SIGN LANGUAGE RECOGNITION USING PYTHON

MINOR PROJECT REPORT

Submitted for the partial fulfillment of the degree of

Bachelor of Technology

In

Internet of Things (IOT)

Submitted By

Brajnandan patidar
0901IO221024

UNDER THE SUPERVISION AND GUIDANCE OF

Dr.Soumayjit Ghosh
Assistant professor

Centre for Internet of Things

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR (M.P.), INDIA
माधव प्रौद्योगिकी एवं विज्ञान संस्थान, ग्वालियर (म.प्र.), भारत
Deemed to be university
NAAC ACCREDITED WITH A++ GRADEA

DECLARATION BY THE CANDIDATE

I hereby declare that the work entitled **"SIGN LANGUAGE RECOGNITION USING PYTHON"** is my work, conducted under the supervision of **Dr.Soumayjit Ghosh, Assistant professor,** during the session May-Dec 2024. The report submitted by me is a record of bonafide work carried out by me.

I further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Brajnandan patidar
0901IO221024

B.Tech IOT V Sem

Date: 19-11-24
Place: Gwalior

This is to certify that the above statement made by the candidates is correct to the best of my knowledge and belief.

Guided By:
Soumayajit Ghosh 19/11/24
Dr.Soumayjit Ghosh
Assistant professor
Centre for Internet of Things
MITS, Gwalior

Departmental Project Coordinator
Dr. Nookala venu
Assistant Professor
Centre for Internet of Things
MITS, Gwalior

Approved by HoD
19/11/2024
Dr. Praveen Bansal
Assistant Professor
Centre for Internet of Things
MITS, Gwalior

ABSTRACT

Interacting with individuals who have hearing or speech impairments often poses significant challenges. Sign language has undoubtedly become an essential medium for such individuals to express their thoughts and emotions, facilitating their integration into society. However, sign language alone is not sufficient. A major limitation is that the signs can be misinterpreted or confusing for those who are unfamiliar with it or have learned a different variation of the language.

This project focuses on creating a machine learning model capable of recognizing and classifying hand gestures used in fingerspelling for sign language. The model is designed to function effectively in diverse environments, accommodating a wide range of users and conditions. As a user-independent model, it trains classification algorithms on one dataset and tests them on an entirely separate dataset.

In this project, we introduce a Sign Language Recognition System utilizing American Sign Language (ASL). Through this system, users can employ a webcam or camera to identify sign language gestures in real time.

.This project is centered on developing a machine learning model capable of identifying and categorizing hand gestures utilized in fingerspelling for sign language. The model is meticulously crafted to operate proficiently in diverse settings, catering to a broad spectrum of users and conditions. Functioning as a user-independent model, it trains classification algorithms on one dataset and tests them on an entirely distinct dataset.

Within this endeavor, we introduce a Sign Language Recognition System that employs American Sign Language (ASL). Through this system, users can utilize a webcam or camera to discern sign language gestures in real-time.

To accomplish this, we processed images of ASL gestures using various computer vision methodologies and crafted a machine learning model. The project harnesses deep learning neural networks, which seek to emulate human cognitive processes through the utilization of data inputs, weights, and biases. These networks comprise multiple layers of interconnected nodes that refine and optimize predictions or classifications. The process, termed forward propagation, commences at the input layer—where the model acquires data—and culminates at the output layer, which furnishes the ultimate prediction or classification.

PLAGIARISM CHECK CERTIFICATE

This is to certify that I/we, a student of B.Tech. in **Internet of Things (IOT)** have checked my complete report entitled "" for similarity/plagiarism using the "Turnitin" software available in the institute.

This is to certify that the similarity in my report is found to be .18%.,which is within the specified limit (20%).

The full plagiarism report along with the summary is enclosed.

Braj

Brajnandan patidar
0901IO221024

Soumyajith Ghosh 19/11/24

Checked & Approved By:

Dr.Soumyajit Ghosh

Centre for Internet of Things
MITS, Gwalior

CONTENT

LIST OF FIGURES:

CHAPTER 1: INTRODUCTION

Language plays a fundamental role in human interaction, serving as the primary means for communication. Mastering a language involves learning an intricate system of vocabulary, grammar, and structure to effectively convey ideas. For most individuals, this skill develops naturally, even before they begin speaking. As people grow, they refine their linguistic abilities, utilizing language to articulate complex thoughts. Embedded in our daily lives, language's importance often goes unnoticed until its absence is felt.

Unfortunately, in today’s fast-paced world, individuals with hearing or speech impairments are frequently marginalized, leaving them excluded from many societal interactions.

Why Choose Sign Language?

Sign language is a form of communication tailored for individuals who are deaf or hard of hearing. This gesture-based system allows for effective communication by transcending the limitations of auditory impairments. Despite being a powerful tool for the deaf community, sign language poses significant challenges for non-signers, further exacerbating the communication gap.

The Need for a Sign Language Recognition System

A key issue with sign language is its limited accessibility to the general population. Like any spoken language, learning sign language requires substantial time and effort, deterring many from acquiring it. This creates a significant barrier for seamless interaction between signers and non-signers.

Machine learning and image recognition offer promising solutions to address this gap. By developing predictive models to automatically classify sign language gestures, we can create tools for real-time translation. Such systems could enable features like live captions for virtual meetings, fostering two-way communication. This would greatly empower individuals with hearing or speech impairments while helping non-signers understand their messages effortlessly.

How Sign Language Recognition Works

Sign language is a visual mode of communication involving hand gestures, facial expressions, and body movements. For example, to represent "deer" in American Sign Language (ASL), one mimics antlers by spreading fingers near the head. Similarly, the sign for "eat" involves bringing the hand toward the mouth, mimicking the act of eating.

Finger spelling, the use of hand signs for individual letters, is crucial for spelling out names or unfamiliar words. ASL, for instance, employs single-handed signs for all alphabet letters. While finger spelling enhances communication, it can be challenging for beginners due to the speed and precision required.

Context, inflection, and modifications further enrich sign language. For instance, adding repetition or altering the speed of a sign can indicate tense or emphasis. Nonmanual markers, such as eyebrow movements or head tilts, also play a vital role in conveying meaning. This combination of manual and nonmanual elements makes sign language both expressive and complex.

Benefits of Sign Language

Proficiency in sign language enhances active listening skills. Maintaining eye contact, a necessity in sign communication, fosters genuine engagement. This habit translates well into spoken conversations, promoting attentiveness.

Additionally, sign language offers practical benefits in situations where speech is impractical, such as underwater communication, noisy environments, or discreet conversations.

Problem Statement

Developing software solutions to bridge communication gaps for individuals with hearing impairments is a critical area of research. Automated Sign Language Recognition (ASLR) systems can facilitate interactions between signers and non-signers, provide educational tools, and serve as assistive technologies.

However, despite progress, ASLR remains a developing field. Drawing parallels with Automatic Speech Recognition (ASR), ASLR systems need further refinement to achieve high accuracy and reliability under varied conditions. This project aims to address some of these challenges and propose innovative solutions.

CHAPTER 2: LITERATURE SURVEY

Early research in vision-based sign language recognition systems was pioneered by Starner and Pentland, who utilized a single video camera paired with uniformly colored gloves to simplify segmentation and feature extraction processes. They later demonstrated that a user-calibrated skin color model could achieve comparable results in controlled environments 【1】.

Hienz et al. introduced the use of color-coded gloves to extract detailed finger information from the dominant hand. Their approach was limited to a plain white background, as arbitrary environments could introduce challenges where skin color or other features might overlap with non-hand elements. This highlighted the insufficiency of relying solely on color information, even when using gloves【2】.

Siming He proposed a system employing a dataset of 40 commonly used words with 10,000 sign language images. Faster R-CNN with an embedded Region Proposal Network (RPN) was utilized to locate hand regions in video frames, outperforming single-stage detection methods like YOLO in terms of accuracy. The study achieved a recognition rate of 99% on a common vocabulary dataset by combining hand localization, 3D CNN for feature extraction, and LSTM coding for sequence recognition【3】.

In another study, a low-cost method for image processing was demonstrated using a green background to simplify RGB to black-and-white conversion. This approach mapped gestures in Sinhala sign language using centroid methods, achieving 92% accuracy in gesture recognition despite hand size or position variations【4】.

Geetha and Manjusha worked on Indian Sign Language recognition using B-spline approximations for alphabets and numerals. By isolating gesture regions and transforming boundaries into smooth B-spline curves, they extracted features for classification via a Support Vector Machine (SVM), attaining an accuracy rate of 90%【5】.

Pigou applied convolutional neural networks (CNNs) to a dataset of 20 Italian sign gestures, utilizing 2D kernels across six layers. Feature extraction was managed by CNNs, and classification was performed using Artificial Neural Networks (ANNs), yielding an accuracy of 91.7%【6】. Similarly, Huang utilized a 3D CNN model, incorporating channels for color and depth alongside body skeleton data, achieving a 94.2% average accuracy across 25 commonly used vocabulary items【8】.

Rumana et al. created a real-time prototype for recognizing ASL alphabets, testing the validity of vision-based recognition systems for real-time applications. The study highlighted the potential for using machine learning algorithms to enable efficient sign language recognition【11】.

A dataset for Indian Sign Language was developed using smartphone front cameras. Processing included pre-filtering, segmentation, and feature extraction to create a sign feature space for classification. The system achieved a Word Matching Score (WMS) of 90.58% using techniques like enhanced Sobel edge detection【12】.

Li et al. addressed scalability in Continuous Sign Language Recognition (CSLR) using Hidden Markov Models (HMMs). Their approach modeled transitions between signs, achieving a real-time word accuracy rate of 87.4% over 1,024 testing sentences【15】.

Another notable effort by Chong and Lee utilized Leap Motion Controllers (LMC) to distinguish between static and dynamic ASL gestures. Combining SVMs and Deep Neural Networks (DNNs), their system achieved recognition rates of 80.3% and 93.81% for static letters and dynamic gestures, respectively【16】.

In summary, the surveyed studies demonstrate diverse approaches to sign language recognition, leveraging advancements in computer vision, machine learning, and deep learning. However, challenges such as scalability, accuracy in diverse environments, and real-time processing remain key areas for future exploration.

CHAPTER 3: BACKGROUND RESEARCH

What is Sign Language?

Sign language is a visual communication system utilized by the hearing-impaired community. For individuals born with hearing impairments, speaking becomes challenging due to the inability to hear and mimic sounds. Instead, they rely on alternative communication methods, using their hands, facial expressions, and body language to express ideas and emotions.Hand gestures are used to depict shapes, actions, and directions. For example, pointing from one location to another visually represents movement. Similarly, facial expressions convey emotional nuances, complementing the message communicated by hand gestures. Thus, sign language serves as an effective communication tool for individuals with hearing impairments.

Interestingly, the use of gestures is not limited to the hearing-impaired community. For instance, babies and toddlers use gestures to communicate before they develop speech. Adults also rely on gestures in scenarios where verbal communication is impractical, such as in noisy environments or underwater. However, the signs used by the hearing-impaired are structured and adhere to specific grammatical rules, forming what is formally recognized as sign language.

According to the World Federation of the Deaf (WFD), there are approximately 70 million deaf individuals worldwide. Contrary to popular belief, sign language is not universal. There are hundreds of unique sign languages globally, each tied to the culture and community of the users. Like spoken languages, sign languages are natural languages with their own grammar and vocabulary, capable of conveying abstract and complex meanings.

Composition of Signs

Signs in any sign language involve two major components: **manual features** and **non-manual features**.

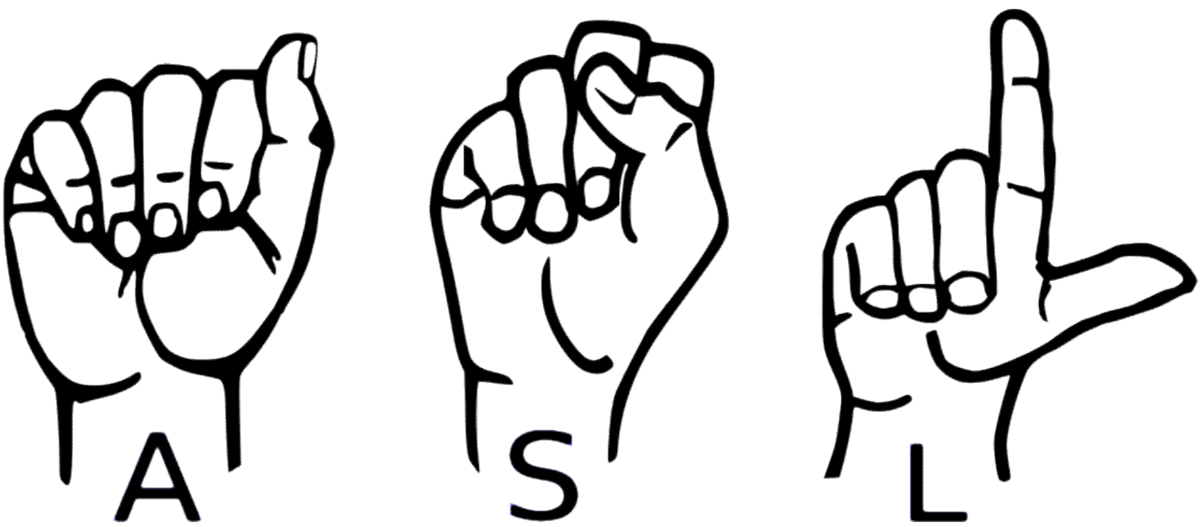
- **Manual Features** include hand shape, orientation, movement, and location.
- **Non-Manual Features** involve facial expressions, head movements, and body postures, which often convey grammatical information.

Three universal rules govern the composition of signs across different sign languages:

Signing Space
Signs are performed within a defined region around the upper body, known as the signing space (Fig. 3.2).

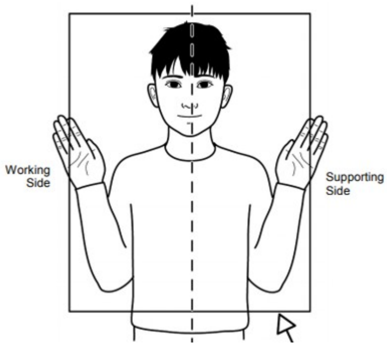
Symmetry Rule
If both hands are involved in a sign and move simultaneously, they must use the same hand shape (Fig. 3.3).

Dominance Rule
When two hands with different shapes are used, the dominant hand performs the movement while the passive hand remains stationary (Fig. 3.4).



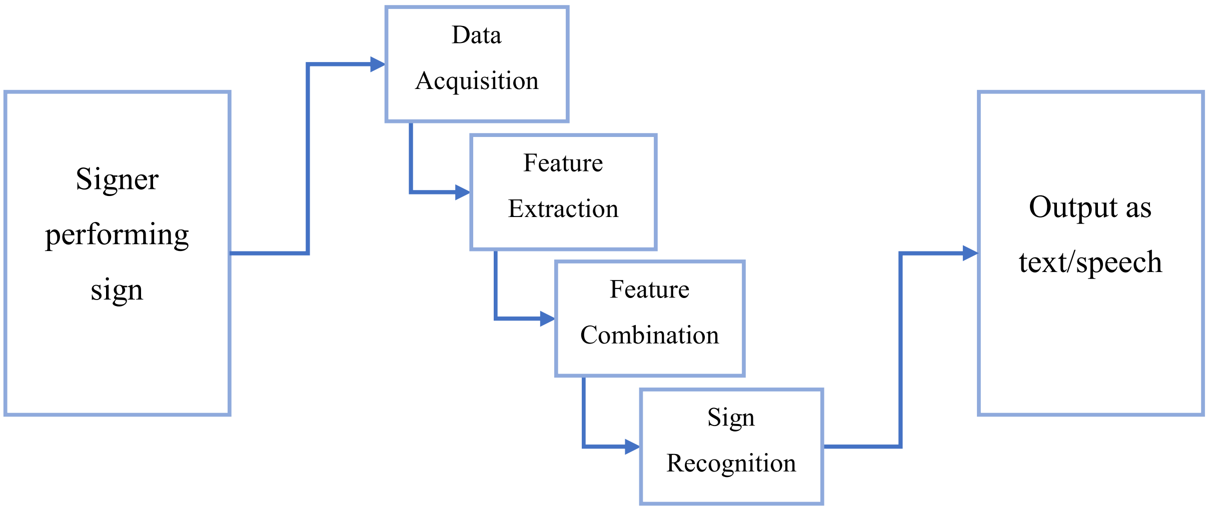
History of Sign Language

The history of sign language reveals a journey of struggle and recognition. In ancient times, Aristotle incorrectly theorized that learning was possible only through hearing spoken words, leading to the widespread misconception that deaf individuals were incapable of education.



This belief denied them basic rights, such as property ownership and marriage.

During the Renaissance, European scholars challenged these notions, marking the beginning of formalized sign language. Over time, notable deaf individuals demonstrated their capabilities, dispelling myths about their limitations.



Sign Language Recognition (SLR)

SLR refers to the automated interpretation of sign language gestures into text or speech without human intervention. The process involves:

Data Acquisition
Collecting video or sensor data of a person signing.

Feature Extraction
Identifying and isolating manual and non-manual features.

Feature Combination
Analyzing the extracted features to interpret the sign's meaning.

Output Generation
Presenting the recognized sign as text or speech.

CHAPTER 4:METHODOLOGY

American Sign Language Recognition

American Sign Language (ASL) consists of two main components: **manual gestures** and **non-manual elements**. Manual gestures involve the hands and arms, forming the core of ASL's lexical meaning. Non-manual elements, such as facial expressions, head movements, and body posture, add contextual details like intonation, intensity, or tense.

While an ideal Automated Sign Language Recognition (ASLR) system would incorporate both components, capturing non-manual gestures accurately, especially in mobile environments, presents significant challenges. Despite this, a system focusing solely on manual gestures can still address many practical applications.

The core attributes of manual signs include:

- **Shape**: The configuration of fingers and palms.
- **Orientation**: The direction the palm or fingers face.
- **Location**: The position of the hands relative to the body.
- **Movement**: The motion of the hands or fingers during signing.

To ensure accurate recognition, an ASLR system must account for both static and dynamic elements of signs. Additionally, transitional movements between signs, akin to coarticulation in spoken language, must be ignored to prevent misclassification. Complexities such as hand overlap or occlusion must also be addressed to ensure robustness.

Capturing and Classifying Sign Gestures

The two primary methods for capturing hand gestures are **computer vision** and **direct sensor-based capture** (e.g., gloves). While sensor-based methods can provide precise data, computer vision is more desirable due to its accessibility and lack of dependency on specialized equipment.

Hand Tracking

Effective hand tracking requires capturing the signer’s upper body in the camera's field of view. While 2D video requires controlled backgrounds, 3D stereo cameras can overcome these limitations, albeit at a higher computational cost.

Feature Extraction

Key features, such as hand position, shape, orientation, and motion trajectories, are critical for accurate classification. Detailed discussions on these methods are available in surveys like Ong and Ranganathan's comprehensive review【27】.

Classification
Sign gestures can be classified either holistically or by breaking them into components. Neural networks are particularly effective for static signs, while Hidden Markov Models (HMMs) excel in recognizing dynamic, sequential signs due to their suitability for time-series data.

Questions and Objectives

The primary objective of this project is to develop a proof-of-concept ASLR system leveraging cloud resources. It aims to demonstrate and evaluate various architectural strategies for building such a system.

The prototype has undergone significant improvements:

- Earlier iterations analyzed every pixel in an image, resulting in high resource consumption, noise interference, and low accuracy.
- The current version focuses on extracting and analyzing only key hand landmarks, significantly improving efficiency and accuracy.

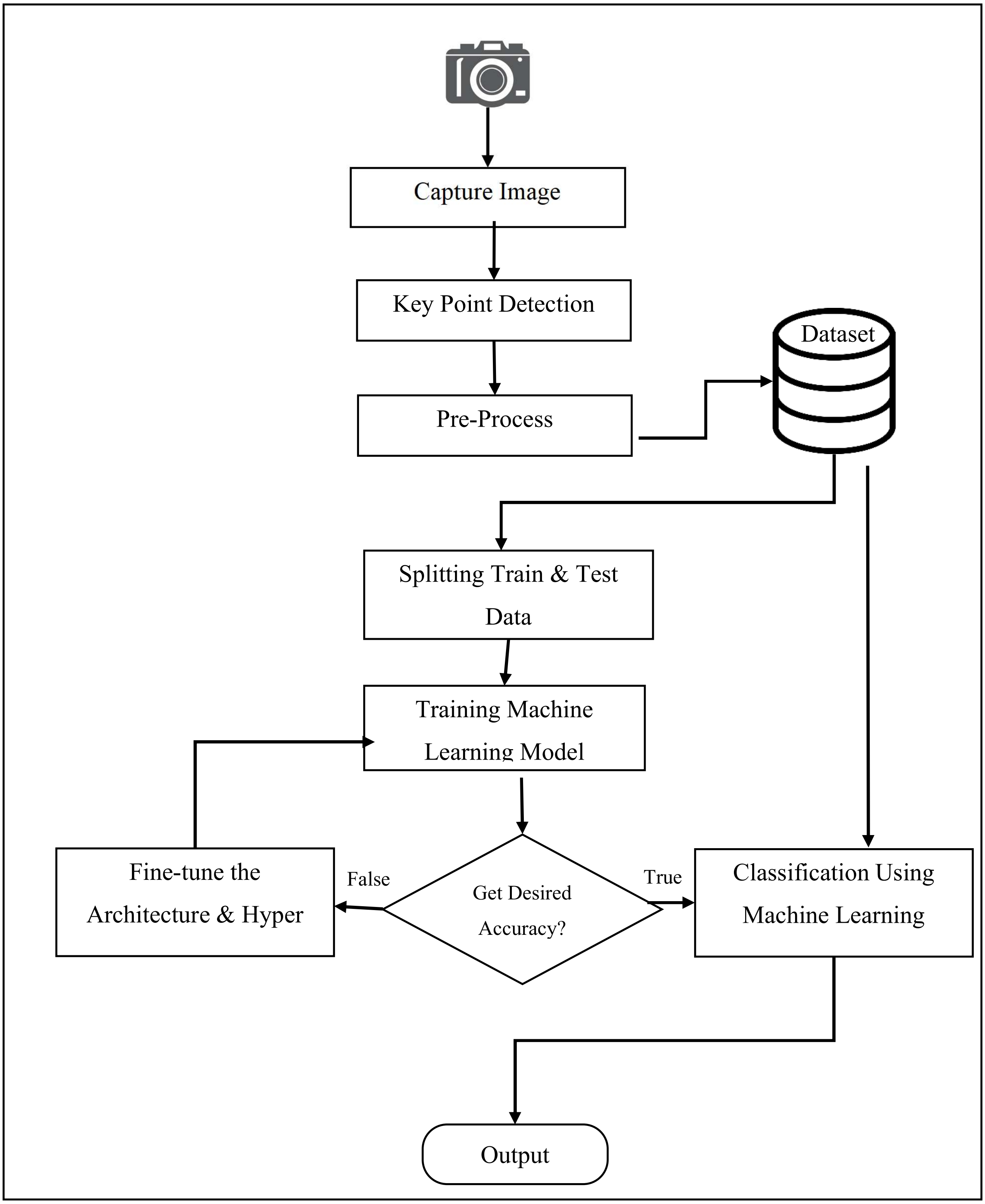
The system's user-friendly graphical interface ensures ease of operation, making it accessible for non-technical users.

Flowchart Overview

The system workflow follows a structured pipeline (Fig. 4.1):

1. **Image Capture**: Capturing hand gestures via camera.
2. **Data Preprocessing**: Extracting and normalizing key landmarks from images.
3. **Dataset Creation**: Splitting data into training and testing sets.
4. **Model Training**: Fine-tuning the architecture and hyperparameters of a machine learning model.
5. **Testing Phase**: Using the trained model to recognize gestures in real-time.
6. **Output Generation**: Translating recognized gestures into text or speech.

Flow Chart:



Capture Image
Key Point Detection
Pre-Process
Dataset
Splitting Train & Test Data
Training Machine Learning Model
Fine-tune the Architecture & Hyper
False
Get Desired Accuracy?
True
Classification Using Machine Learning
Output

CHAPTER 5: SYSYTEM REQUIREMENT

This section outlines the minimum hardware, software, and external resources needed for the development and operation of the Sign Language Recognition System. These requirements ensure smooth execution while allowing for potential upgrades.

Hardware Requirements (Minimum)

- **Processor**: Intel Core i5 or AMD Ryzen 5
- **RAM**: 8 GB
- **Storage**: 2.5 GB of free space
- **Graphics**: Integrated Graphics Card
- **Input Devices**: Keyboard and mouse
- **Output Devices**: Monitor
- **Capture Device**: External webcam (if the system lacks a built-in camera)

Software Requirements (Minimum)

- **Programming Language**: Python 3.10
- **Operating System**:
  - Windows 7 or higher
  - Linux
  - macOS 10.12.6 or higher (64-bit)
- **Integrated Development Environment (IDE)**:
  - VS Code, PyCharm, or any other compatible IDE

External Dependencies

The system relies on the following Python modules:

1. **TensorFlow (v2.12.0)** – Core framework for deep learning.
2. **Keras (v2.12.0)** – High-level API for neural network creation.
3. **MediaPipe (v0.10.0)** – For detecting hand landmarks.
4. **Scikit-learn (v1.2.2)** – For data preprocessing and model evaluation.
5. **Pandas (v2.0.1)** – Data manipulation and analysis.
6. **NumPy (v1.24.3)** – Numerical operations and array handling.
7. **OpenCV (v4.6.0.66)** – Computer vision techniques for image processing.
8. **Matplotlib (v3.7.1)** – Visualization of data and results.

This combination of hardware and software ensures that the system can handle data acquisition, processing, model training, and real-time testing with minimal resource consumption.

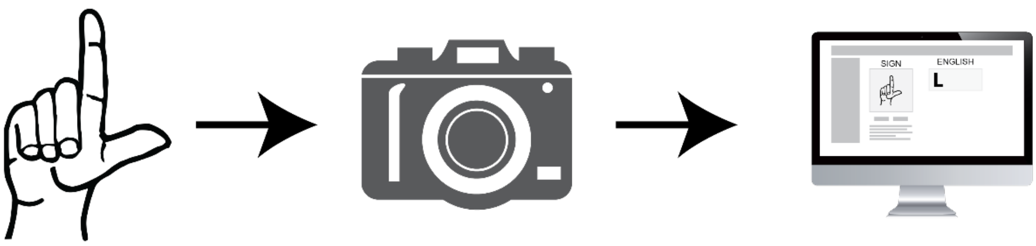
CHAPTER 6: SYSTEM ARCHITECTURE

Overall Architecture

From a physical standpoint, the system comprises two main components:

1. **Capture Devices**: These devices (e.g., webcams) observe the signing individual and capture the required visual information.
2. **Computing Unit**: This processes the captured data, analyzes the gestures, and identifies their corresponding meanings.

The interaction between these components is illustrated in **Fig. 6.1**.



Performed Sign — Capturing Image — Predicting Sign

Component Design

The system consists of three primary components:

Presentation Layer (UI)

1. This layer interacts with the user, providing a clear and accessible interface.
2. It displays the recognized signs as text or speech, offering multiple output formats.

Hand Component

1. This module focuses on manual gestures, storing information about hand shapes and positions.
2. It enables hand shape recognition, a critical step in interpreting gestures.

3. The hand recognition functionality is powered by **MediaPipe**, which detects and processes 26 selected hand shapes corresponding to the ASL alphabet.

Recognition Component

1. This module analyzes the data from the hand component to identify performed signs.
2. It compares hand shapes to a predefined dictionary of gestures, focusing on the ASL alphabet.
3. The component outputs the recognized signs in real-time, translating them into text and speech.

Key Features of the Recognition Process

Manual Features Only:
The project emphasizes manual gestures, leaving non-manual features for future iterations. Manual gestures provide the core structure for recognition and are sufficient for the intended scope of this system.

Scalability:
While the current system is limited to 26 ASL alphabet signs, the architecture can be expanded to accommodate additional gestures and complex sign compositions.

Output Formats:
Recognized gestures are presented in two formats:

-
  - Text: Displayed on the system's terminal.
  - Speech: Generated through text-to-speech functionality for auditory output.

The modular design of the system ensures flexibility, allowing for future enhancements such as dynamic sign recognition and support for more complex languages.

CHAPTER 7: SYSTEM IMPLEMENTATION

This section details the implementation of the Sign Language Recognition System, which uses Python and deep learning techniques to capture, process, and classify hand gestures. The system operates through a structured workflow involving data acquisition, preprocessing, model training, and real-time prediction.

Machine Learning Overview

Machine learning (ML) enables computers to learn and improve from data without explicit programming. The system uses ML algorithms to create a model based on labeled training data, which is then used to classify or predict outcomes for new data.

ML applications span various domains, such as speech recognition, computer vision, and medical diagnostics, where traditional programming is insufficient or impractical.

Supervised Learning

This project employs supervised learning, where labeled data is provided to the system for training. The model maps inputs to their corresponding outputs by learning patterns in the data. The system's accuracy is validated by testing it on new data to ensure reliable predictions.

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks are a type of deep learning architecture well-suited for image recognition tasks. Key components of CNNs include:

- **Convolutional Layers**: Extract features from input images.
- **Pooling Layers**: Reduce the spatial dimensions of data, preserving key features while minimizing computational complexity.
- **Fully Connected Layers**: Combine extracted features to make final classifications.

Hand Tracking

Image Capture
The system uses OpenCV to capture live images of hand gestures through a webcam. Images are converted from BGR to RGB format for processing.

Key Point Detection
With MediaPipe, 21 landmarks (key points) are detected for each hand (Fig. 7.2). These landmarks, represented as (x, y) coordinates, define the hand's shape and position.

Debugging and Visualization
During hand detection, a visual overlay displays the detected landmarks and labels on the screen (Fig. 7.3). This aids users in verifying the accuracy of hand

detection.

Index	0.x	0.y	1.x	1.y	1.x	2.y	2.x	3.y	4.x	4.y	5.x	5.y	6.x	6.y	7.x	7.y	8.x	8.y	9.x
0	0	0	-0.2605	-0.12605	-0.48739	-0.42017	-0.59664	-0.67227	-0.48739	-0.83193	-0.36134	-0.77311	-0.36134	-1	-0.34454	-0.7395	-0.33613	-0.60504	-0.15126
0	0	0	-0.24576	-0.15254	-0.4661	-0.42373	-0.60169	-0.67797	-0.5	-0.85593	-0.33898	-0.76271	-0.35593	-1	-0.33051	-0.74576	-0.31356	-0.59322	-0.12712
0	0	0	-0.24167	-0.11667	-0.475	-0.41667	-0.6	-0.68333	-0.49167	-0.86667	-0.36667	-0.75833	-0.38333	-1	-0.35833	-0.725	-0.34167	-0.59167	-0.15833
0	0	0	-0.23729	-0.13559	-0.4661	-0.41525	-0.60169	-0.66949	-0.5	-0.84746	-0.35593	-0.75424	-0.36441	-1	-0.33051	-0.73729	-0.32203	-0.58475	-0.13559
0	0	0	-0.26271	-0.12712	-0.49153	-0.39831	-0.61017	-0.66102	-0.52542	-0.85593	-0.38983	-0.74576	-0.38136	-1	-0.34746	-0.74576	-0.33898	-0.58475	-0.16949
1	0	0	-0.15	-0.10909	-0.22273	-0.28636	-0.18182	-0.45455	-0.08182	-0.52273	-0.20455	-0.51364	-0.23182	-0.70909	-0.23636	-0.82727	-0.23182	-0.92727	-0.1
1	0	0	-0.15421	-0.09813	-0.23364	-0.27103	-0.18692	-0.42523	-0.09346	-0.50467	-0.21028	-0.50935	-0.24299	-0.70561	-0.25234	-0.8271	-0.24299	-0.92991	-0.1028
1	0	0	-0.15814	-0.09767	-0.23256	-0.27907	-0.19535	-0.44186	-0.10233	-0.50698	-0.2093	-0.52093	-0.24186	-0.71163	-0.25116	-0.82791	-0.24651	-0.93023	-0.10698
1	0	0	-0.15138	-0.09633	-0.22477	-0.27064	-0.19266	-0.42661	-0.12385	-0.50917	-0.20642	-0.51376	-0.23853	-0.70642	-0.24312	-0.8211	-0.23853	-0.92202	-0.09633
1	0	0	-0.15668	-0.09677	-0.23041	-0.27189	-0.20737	-0.43318	-0.14286	-0.51613	-0.20276	-0.51613	-0.23963	-0.70968	-0.24885	-0.82488	-0.24885	-0.93088	-0.10138
2	0	0	-0.28467	-0.07299	-0.51825	-0.19708	-0.71533	-0.24818	-0.86861	-0.35766	-0.40876	-0.64234	-0.52555	-0.90511	-0.67153	-0.91241	-0.78832	-0.84672	-0.32117
2	0	0	-0.28058	-0.07194	-0.5036	-0.20863	-0.69784	-0.26619	-0.83453	-0.38129	-0.40288	-0.64748	-0.52518	-0.89928	-0.67626	-0.91367	-0.78417	-0.84892	-0.31655
2	0	0	-0.26619	-0.06475	-0.48201	-0.21583	-0.67626	-0.27338	-0.79137	-0.38849	-0.38129	-0.65468	-0.4964	-0.91367	-0.64029	-0.92086	-0.7482	-0.84892	-0.29496
2	0	0	-0.27536	-0.06522	-0.5	-0.2029	-0.69565	-0.26812	-0.82609	-0.3913	-0.39855	-0.64493	-0.51449	-0.9058	-0.66667	-0.92754	-0.77536	-0.86232	-0.31159
2	0	0	-0.27536	-0.06522	-0.50725	-0.21014	-0.7029	-0.26812	-0.81159	-0.39855	-0.3913	-0.65217	-0.51449	-0.9058	-0.66667	-0.92754	-0.78261	-0.86232	-0.30435
3	0	0	-0.15979	-0.10825	-0.26804	-0.29381	-0.30928	-0.45876	-0.24742	-0.58247	-0.2268	-0.52577	-0.29381	-0.74227	-0.34536	-0.87629	-0.37113	-1	-0.1134
3	0	0	-0.1641	-0.10769	-0.26667	-0.30256	-0.31282	-0.46154	-0.25128	-0.58462	-0.23077	-0.52821	-0.29744	-0.74872	-0.34359	-0.88205	-0.36923	-1	-0.11795
3	0	0	-0.1641	-0.11282	-0.26667	-0.30769	-0.31282	-0.46667	-0.25641	-0.59487	-0.2359	-0.52821	-0.30256	-0.74872	-0.34872	-0.88205	-0.37436	-1	-0.12308
3	0	0	-0.15464	-0.10825	-0.26289	-0.30412	-0.31443	-0.45876	-0.26289	-0.58763	-0.2268	-0.52577	-0.28866	-0.74742	-0.34021	-0.88144	-0.36598	-1	-0.1134
3	0	0	-0.15625	-0.10417	-0.26042	-0.30208	-0.30729	-0.45833	-0.25	-0.57292	-0.22396	-0.52604	-0.29167	-0.75	-0.34375	-0.88021	-0.36458	-1	-0.10938
4	0	0	-0.27083	-0.09722	-0.4375	-0.33333	-0.40278	-0.56944	-0.23611	-0.66667	-0.43056	-0.60417	-0.5	-0.86111	-0.52083	-0.75694	-0.47917	-0.61806	-0.27778
4	0	0	-0.25874	-0.08392	-0.41958	-0.34965	-0.37063	-0.58042	-0.21678	-0.67133	-0.41958	-0.61538	-0.4965	-0.87413	-0.5035	-0.78322	-0.46154	-0.64336	-0.27273
4	0	0	-0.25	-0.09722	-0.40972	-0.33333	-0.375	-0.56944	-0.21528	-0.67361	-0.40972	-0.61111	-0.47917	-0.875	-0.49306	-0.76389	-0.45139	-0.61806	-0.26389
4	0	0	-0.25874	-0.09091	-0.42657	-0.34965	-0.38462	-0.58741	-0.2028	-0.66434	-0.42657	-0.60839	-0.4965	-0.87413	-0.5035	-0.76923	-0.45455	-0.62937	-0.27972
4	0	0	-0.25	-0.09028	-0.42361	-0.34028	-0.38889	-0.58333	-0.20833	-0.65278	-0.41667	-0.61111	-0.48611	-0.86111	-0.49306	-0.74306	-0.45139	-0.61111	-0.27083
5	0	0	-0.24051	-0.07595	-0.44304	-0.20886	-0.56962	-0.36709	-0.57595	-0.53797	-0.37975	-0.46835	-0.50633	-0.62025	-0.5443	-0.61392	-0.5443	-0.56329	-0.26582
5	0	0	-0.24684	-0.06329	-0.4557	-0.20886	-0.59494	-0.36709	-0.60127	-0.53797	-0.39873	-0.46203	-0.52532	-0.62025	-0.56962	-0.62658	-0.57595	-0.58228	-0.27848
5	0	0	-0.23899	-0.06289	-0.44654	-0.20126	-0.58491	-0.3522	-0.59119	-0.51572	-0.38365	-0.45912	-0.50943	-0.61635	-0.55346	-0.61006	-0.56604	-0.55975	-0.27044
5	0	0	-0.24359	-0.05128	-0.46154	-0.19231	-0.59615	-0.34615	-0.60256	-0.51923	-0.39744	-0.44872	-0.51923	-0.59615	-0.57051	-0.60256	-0.57692	-0.5641	-0.28205

Landmark Calculation
The coordinates of the detected landmarks are stored in a structured format for further analysis. The system transforms the data into an array for compatibility with machine learning models (Fig. 7.4).

Data Preprocessing
Landmarks are normalized to relative coordinates, making them independent of the frame’s dimensions. The 2D array is flattened into a 1D array, with values scaled between 0 and 1 to standardize the dataset (Fig. 7.5).

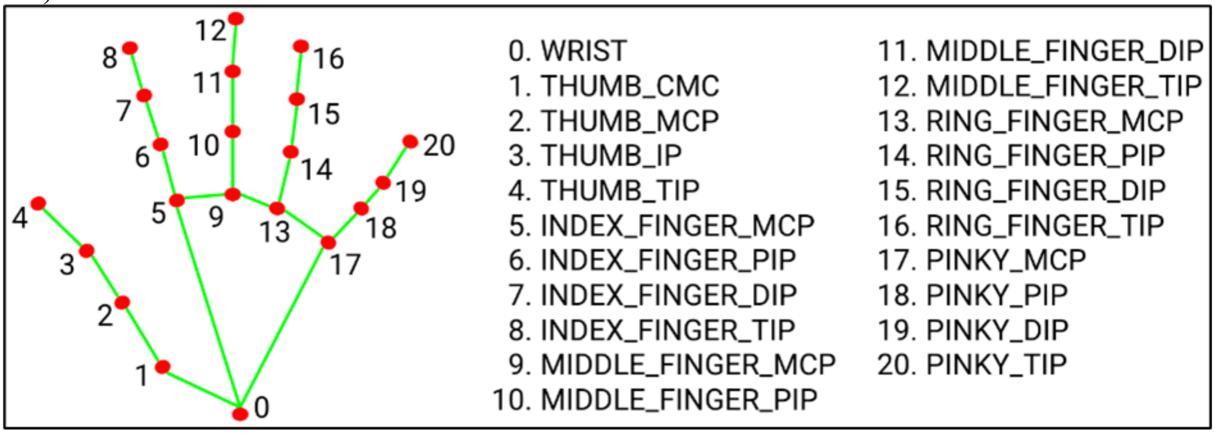
Data Collection

The system provides two modes:

1. **Prediction Mode**: Recognizes gestures in real time.
2. **Logging Mode**: Captures and stores data for model training.

When in logging mode, the system saves preprocessed landmarks along with their corresponding labels in a CSV file for creating a dataset (Fig.

7.6).



Training Phase

The model is trained using the **Keras Sequential API** from TensorFlow. Key components include:

- **Input Layer**: Accepts 42 features (21 normalized landmarks for each hand).
- **Hidden Layers**: Two fully connected layers with ReLU activation and dropout for regularization.
- **Output Layer**: Uses softmax activation to classify gestures into one of 24 ASL letters.

Training Details:

- **Loss Function**: Sparse Categorical Crossentropy (SCCE), which is efficient for mutually exclusive classes.
- **Optimizer**: Adam, chosen for its efficiency and adaptability.
- **Validation**: Data is split into training and testing sets to evaluate model performance.

The model's architecture is summarized in Fig. 7.8.

Testing Phase

In the testing phase, the system processes live camera feed to detect gestures. Key steps include:

1. Capturing hand landmarks.
2. Preprocessing data using the same normalization techniques as during training.

3. Feeding data to the trained model for classification.
4. Displaying the recognized gesture as text or speech output.

Real-time predictions are visualized on the interface, allowing users to see the system's outputs instantly (Fig. 7.9).

```
... Model: "sequential"
    _________________________________________________________________
     Layer (type)                Output Shape              Param #
    =================================================================
     dropout (Dropout)           (None, 42)                0

     dense (Dense)               (None, 20)                860

     dropout_1 (Dropout)         (None, 20)                0

     dense_1 (Dense)             (None, 10)                210

     dense_2 (Dense)             (None, 24)                264

    =================================================================
    Total params: 1,334
    Trainable params: 1,334
    Non-trainable params: 0
    _________________________________________________________________
```

RESULT:

Objective 1: Hand Detection and Data Collection

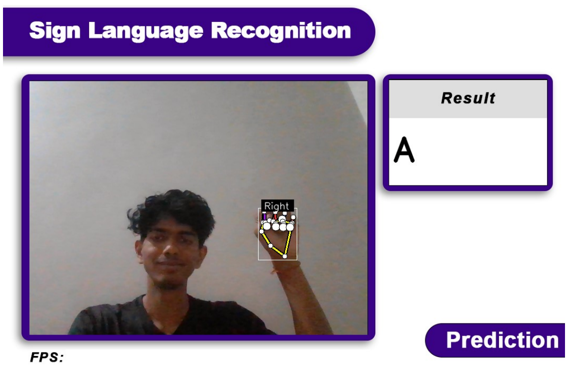
The system effectively detects hand gestures using the following steps:

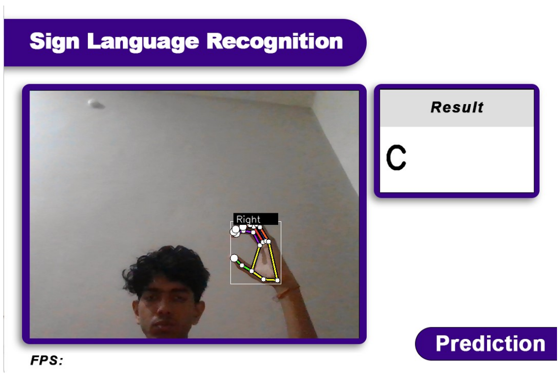
1. **Detection**: Identifying the location and shape of the hand through landmarks.
2. **Preprocessing**: Converting the detected landmarks into a one-dimensional, normalized array for machine learning compatibility.
3. **Storage**: Saving processed data as rows in a CSV file (e.g., keyframe.csv).

Objective 2: Gesture Classification and Recognition

The system was tested for its ability to classify 24 ASL alphabets. Using real-time data, the model successfully matched live hand landmarks with pre-trained data to identify gestures. The following observations were made:

1. **Training Accuracy**: During training, the model achieved an accuracy of 75%.
2. **Real-Time Performance**: Despite the lower training accuracy, real-time predictions were highly reliable for gestures performed correctly by the signer.





Key Insights

Dataset Composition:

- The dataset used for training consisted of 2,400 samples.
- Each sample contained 43 columns: 1 for the label and 42 for the normalized coordinates of the 21 landmarks.

Accuracy Improvements:

- Better image quality, lighting, and processing power improved prediction accuracy.
- Enhanced preprocessing techniques, such as removing noise and refining model parameters, further optimized performance.

```
...  Epoch 1/1000
     132/141 [==========================>..] - ETA: 0s - loss: 3.0102 - accuracy: 0.1074
     Epoch 1: saving model to model\aslr_model.hdf5
     141/141 [==============================] - 1s 3ms/step - loss: 2.9937 - accuracy: 0.1114 - val_loss: 2.6362 - val_accuracy: 0.2240
     Epoch 2/1000
     138/141 [============================>.] - ETA: 0s - loss: 2.5042 - accuracy: 0.2221
     Epoch 2: saving model to model\aslr_model.hdf5
     141/141 [==============================] - 0s 2ms/step - loss: 2.4996 - accuracy: 0.2240 - val_loss: 2.0059 - val_accuracy: 0.5238
     Epoch 3/1000
     138/141 [============================>.] - ETA: 0s - loss: 2.0777 - accuracy: 0.3190
     Epoch 3: saving model to model\aslr_model.hdf5
     141/141 [==============================] - 0s 2ms/step - loss: 2.0736 - accuracy: 0.3200 - val_loss: 1.5257 - val_accuracy: 0.6762
     Epoch 4/1000
     121/141 [========================>.....] - ETA: 0s - loss: 1.8415 - accuracy: 0.3692
     Epoch 4: saving model to model\aslr_model.hdf5
     141/141 [==============================] - 0s 2ms/step - loss: 1.8274 - accuracy: 0.3728 - val_loss: 1.2863 - val_accuracy: 0.7772
     Epoch 5/1000
     136/141 [==========================>..] - ETA: 0s - loss: 1.6941 - accuracy: 0.4107
     Epoch 5: saving model to model\aslr_model.hdf5
     141/141 [==============================] - 0s 2ms/step - loss: 1.6909 - accuracy: 0.4122 - val_loss: 1.1444 - val_accuracy: 0.8108
     Epoch 6/1000
     133/141 [==========================>..] - ETA: 0s - loss: 1.6099 - accuracy: 0.4276
     Epoch 6: saving model to model\aslr_model.hdf5
     141/141 [==============================] - 0s 2ms/step - loss: 1.6068 - accuracy: 0.4282 - val_loss: 1.0486 - val_accuracy: 0.8228
     Epoch 7/1000
     ...
     117/141 [=======================>......] - ETA: 0s - loss: 0.7977 - accuracy: 0.7187
     Epoch 102: saving model to model\aslr_model.hdf5
     141/141 [==============================] - 0s 2ms/step - loss: 0.8030 - accuracy: 0.7186 - val_loss: 0.2998 - val_accuracy: 0.9603
     Epoch 102: early stopping
```

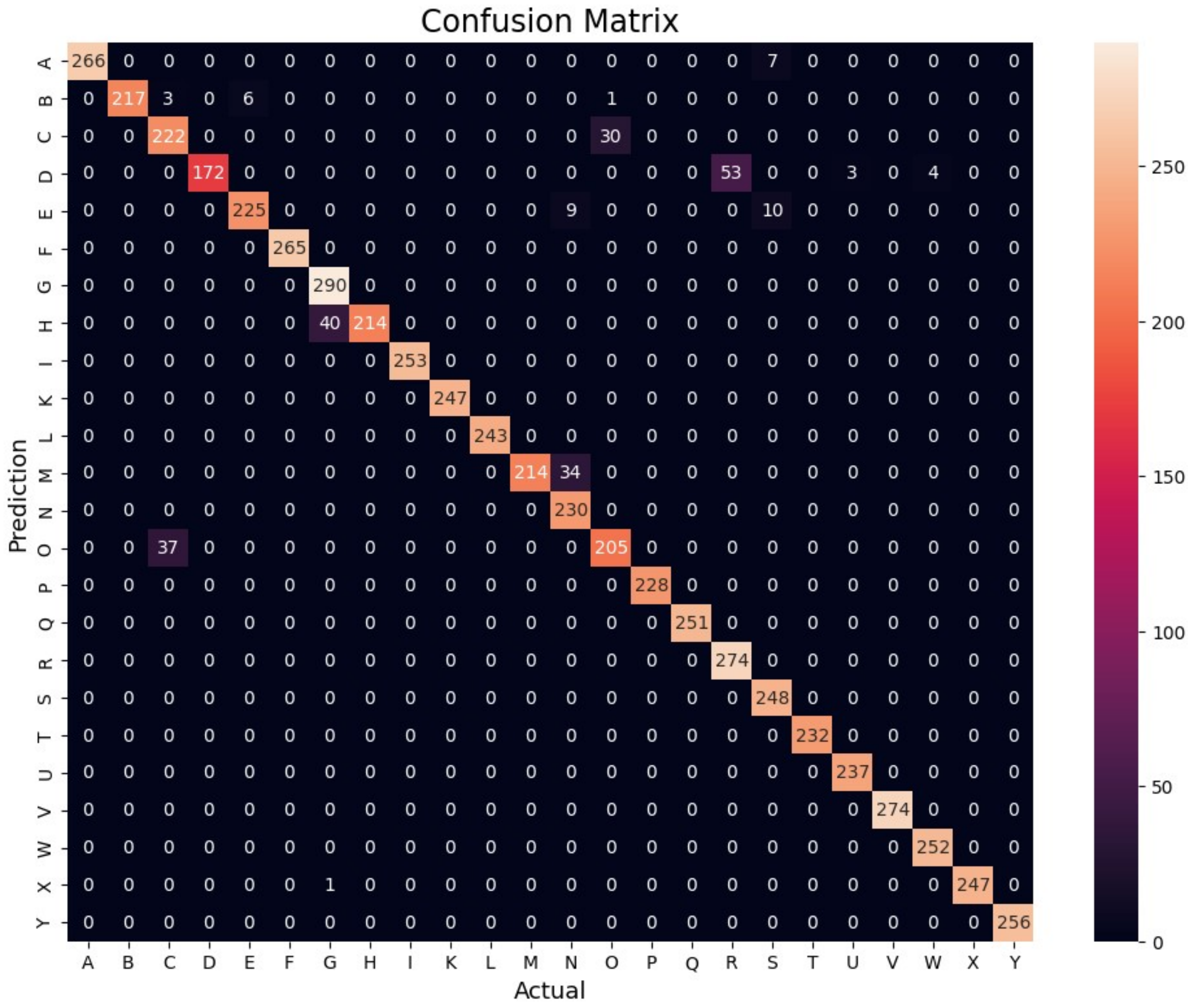
Challenges

- **Low-Quality Input**: Poor lighting or low-resolution images reduced accuracy.
- **Preprocessing Variations**: Inconsistent preprocessing methods impacted prediction reliability.

Through iterative adjustments to activation functions, loss functions, epochs, and other parameters, the system demonstrated improved performance.

Confusion Matrix

A confusion matrix was used to evaluate the system's predictive accuracy across all gesture classes (Fig. 8.2). It highlighted areas where the model performed well and identified classes prone to misclassification.



Model Accuracy

Fig. 8.1 illustrates the model's performance during training and testing phases. The training accuracy stabilized at 71.12%, while testing accuracy reached an impressive 90.6%. These results indicate the system's potential for accurate real-time gesture recognition.

FUTURE SCOPE:

Sign language detection systems hold significant potential for improvement and expanded applications. The following advancements can enhance accuracy, accessibility, and versatility:

1. Improved Recognition Accuracy

- Refining algorithms and models can help minimize errors and increase reliability.
- Incorporating additional training data from diverse users and environments will improve generalizability.

2. Real-Time Gesture Recognition

- Enhancements in computational power and algorithm optimization can enable instantaneous gesture recognition, bridging the gap between signing and spoken communication.

3. Incorporating Non-Manual Features

- Future systems could integrate facial expressions, head movements, and body posture into the recognition process to provide a more holistic understanding of sign language.

4. Integration with Emerging Technologies

- **Augmented Reality (AR)** and **Virtual Reality (VR)** could create immersive communication experiences, such as virtual interpreters or collaborative environments for remote communication.

5. Mobile Accessibility

- Developing lightweight applications for mobile devices can bring sign language recognition to everyday scenarios, empowering users to communicate more effectively on the go

6. Educational Applications

- Detection systems can serve as interactive tools for learning sign language, offering real-time feedback, corrections, and engaging lessons to facilitate language acquisition.

7. Seamless Integration into Digital Platforms

- Integrating sign language detection into video conferencing tools, smart home systems, and online content can enhance inclusivity and accessibility.

---

8. Personalized and Adaptive Systems

- Customizing recognition systems to adapt to individual signing styles or regional variations can provide a more inclusive and personalized user experience.

---

9. Dynamic Gesture Recognition

- Expanding capabilities to recognize dynamic or sequential gestures will open new avenues, such as interpreting full sentences or complex phrases.

---

10. Empowering Accessibility

- Sign language recognition systems have the potential to break communication barriers in education, healthcare, employment, and social interactions. By fostering inclusivity, these systems can contribute to a more equitable society.

The future of sign language detection lies in continuous innovation, collaboration, and commitment to inclusivity. By addressing the outlined areas, these systems can reach their full potential and profoundly impact the lives of millions.

Future Directions

While the current system achieves significant milestones, there is room for improvement:

1. **Expanding Datasets**: Incorporating larger and more diverse datasets will improve accuracy and robustness.

2. **Dynamic Gesture Recognition**: Extending capabilities to include continuous and sequential signing can address more complex communication needs.
3. **Non-Manual Features**: Integrating facial expressions and body language will enhance the system's comprehensiveness.
4. **Enhanced Real-Time Performance**: Optimizing computational efficiency will reduce latency and improve user experience.

In conclusion, this project lays a solid foundation for vision-based sign language recognition systems. With continued innovation and collaboration, it is possible to develop solutions that bridge the communication gap, creating a more inclusive and accessible future for individuals who use sign language

CONCLUSION:

Sign language detection systems possess transformative potential, offering innovative solutions to bridge communication gaps for individuals who use sign language. These systems facilitate seamless interaction between signers and non-signers in various domains, including education, healthcare, and social settings.

Achievements

This project demonstrates a foundational approach to automated sign language recognition:

- The system successfully detects and processes hand gestures, converts them into normalized data, and uses machine learning models to classify gestures in real time.
- With a training accuracy of 71.12% and testing accuracy of 90.6%, the system effectively recognizes 24 ASL alphabet signs, showcasing its capability for practical applications.
- By simplifying the processing pipeline and focusing on critical features, the system is both lightweight and efficient.

Impact and Potential

The development of this system highlights the importance of breaking down communication barriers:

- **Inclusivity**: By enabling individuals who rely on sign language to interact effortlessly with non-signers, the system fosters social equity.
- **Scalability**: The modular design and flexibility of the architecture allow for future expansions, including dynamic gesture recognition and support for complete sign language systems.
- **Accessibility**: With further advancements, this system can serve as a tool for education, professional communication, and personal interactions, empowering users in diverse contexts.

References

1. • Starner, T., Weaver, J., & Pentland, A. *Real-Time American Sign Language Recognition from Video Using Hidden Markov Models*. MIT Media Laboratory, Perceptual Computing Section Technical Report No. 375, 1996.

2. • Hienz, H., & Grobel, K. *Automatic Estimation of Body Regions from Video Images*. In I. Wachsmuth and M. Fröhlich (Eds.), *Gesture and Sign Language in Human-Computer Interaction* (pp. 135–145). Springer-Verlag, Berlin, 1998.

3. • He, S. (2019). *Research of a Sign Language Translation System Based on Deep Learning* (pp. 392–396). DOI: 10.1109/AIAM48774.2019.00083.

4. • Herath, H.C.M., Kumari, W.A.L.V., Senevirathne, W.A.P.B., & Dissanayake, M. (2013). *Image-Based Sign Language Recognition System for Sinhala Sign Language*.

5. • Geetha, M., & Manjusha, U.C. *Vision-Based Recognition of Indian Sign Language Alphabets and Numerals Using B-Spline Approximation*. *International Journal on Computer Science and Engineering (IJCSE)*, 4(3), 406–415, 2012.

6. • Pigou, L., Dieleman, S., Kindermans, P.J., & Schrauwen, B. (2015). *Sign Language Recognition Using Convolutional Neural Networks*. In Agapito, L., Bronstein, M., & Rother, C. (Eds.), *Computer Vision - ECCV 2014 Workshops* (pp. 459–473). Springer, Cham.

7. • Escalera, S., et al. *ChaLearn Looking at People Challenge 2014: Dataset and Results*. *Workshop at the European Conference on Computer Vision* (pp. 459–473). Springer, Cham.

8. • Huang, J., Zhou, W., & Li, H. (2015). *Sign Language Recognition Using 3D Convolutional Neural Networks*. IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6. IEEE.

9. • Mustafa, E. (2014). *Sign Language Interpretation Using Kinect*. MSc Thesis, The University of Sheffield.

10. • Kaggle. *Sign Language MNIST Dataset*. (Accessed July 25, 2022).

11. • Rumana, R., Rani, R.S., & Prema, R. (2021). *A Review Paper on Sign Language Recognition for the Deaf and Dumb*.

12. • Rao, G.A., & Kishore, P.V.V. *Sign Language Recognition System Simulated for Video Captured with Smartphone Front Camera*. *International Journal of Electrical and Computer Engineering (IJECE)*, 6(5), 2176–2187, 2016.

13. • Li, T., et al. *Recognition System for Home-Service-Related Sign Language Using Entropy-Based K-Means Algorithm and ABC-Based HMM*. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(1), 150–162.

14. • Wang, L.C., et al. *Similarity Assessment Model for Chinese Sign Language Videos*. *IEEE Transactions on Multimedia*, 16(3), 751–761, 2014.

15. • Li, K., Zhou, Z., & Lee, C.H. *Sign Transition Modeling and a Scalable Solution to Continuous Sign Language Recognition*. *ACM Transactions on Accessible Computing (TACCESS)*, 8(2), 7–23.

16. • Chong, T.W., & Lee, B.G. *American Sign Language Recognition Using Leap Motion Controller with Machine Learning Approach*. *Sensors*, 18(10), 3554, 2018.

17. • Cardoso, T., Delgado, J., & Barata, J. *Hand Gesture Recognition Towards Enhancing Accessibility*. *6th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Infoexclusion (DSAI 2015)*, 419–429.

18. • Suharjito, R., et al. *Sign Language Recognition Application Systems for Deaf-Mute People: A Review Based on Input-Process-Output*. *Procedia Computer Science*, pp. 441–448, 2017.

19. • Hunter, J.D. *Matplotlib: A 2D Graphics Environment*. *Computing in Science & Engineering*, 9(3), 90–95, 2007.

20. • TensorFlow API Documentation. *Version 2.12.0*. (Accessed July 26, 2022).

21. • Keras API Reference. *Version 2.12.0*. (Accessed July 26, 2022).

22. • MediaPipe Tasks API Reference. (Accessed July 26, 2022).

23. • Scikit-learn API Reference. *Version 1.2.2*. (Accessed July 26, 2022).

TURNITIN PLAGIARISM REPORT

Similarity Report

PAPER NAME	AUTHOR
Brajnandan.pdf	S G

WORD COUNT 5909 Words	CHARACTER COUNT 37196 Characters
PAGE COUNT 38 Pages	FILE SIZE 2.6MB
SUBMISSION DATE Nov 18, 2024 7:46 PM GMT+5:30	REPORT DATE Nov 18, 2024 7:47 PM GMT+5:30

- **18% Overall Similarity**

The combined total of all matches, including overlapping sources, for each database.

- 11% Internet database
- 4% Publications database
- Crossref database
- Crossref Posted Content database
- 16% Submitted Works database

- **Excluded from Similarity Report**

- Bibliographic material
- Small Matches (Less then 10 words)