Report On

Smart Hindi Sign Language Interpreter Using NLP

Submitted in partial fulfillment of the requirements of the Mini project in Semester VI of Third Year Artificial Intelligence & Data Science Engineering

by

Pratik Sanjay Avhad (Roll No. 01)

Priyanka Narendra Bhandari (Roll No. 02)

Swarup Satish Kakade (Roll No. 15)

Guide

Mrs. Sneha Yadav

Co Guide

Ms. Bhavika Gharat

University of Mumbai

Vidyavardhini's College of Engineering & Technology

Department of Artificial Intelligence and Data Science

(A.Y. 2024-25)

Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

CERTIFICATE

This is to certify that the Mini Project entitled **"Smart Hindi Sign Language Interpreter Using NLP"** is a bonafide work of **Pratik Sanjay Avhad (Roll No. 01), Priyanka Narendra Bhandari (Roll No. 02), Swarup Satish Kakade (Roll No. 15),** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **"Bachelor of Engineering"** in Semester VI of Third Year **"Artificial Intelligence and Data Science".**

Co Guide Ms. Bhavika Gharat	Guide Mrs. Sneha Yadav

Ms. Sejal D'mello Deputy HOD AI & DS	Dr. Tatwadarshi Nagarhalli HOD AI &DS	Dr. Rakesh Himte Pribcipal Vcet

Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

Mini Project Approval

This Mini Project entitled "**Smart Hindi Sign Language Interpreter Using NLP" Pratik Sanjay Avhad (Roll No. 01), Priyanka Narendra Bhandari (Roll No. 02), Swarup Satish Kakade (Roll No. 15),** is approved for the degree of **Bachelor of Engineering** in in Semester VI of Third Year **Artificial Intelligence and Data Science.**

Examiners

1…….....
(Internal Examiner Name & Sign)

2……
(External Examiner name & Sign)

Date:

Place:

Acknowledgments

I would like to thank all people whose support and cooperation has been an invaluable asset during this Project. I would also like to thank our Guide Ms. Bhavika Gharat , for guiding me throughout this project and giving it the present shape. It would have been impossible to complete the project without his/her support, valuable suggestions, criticism, encouragement, and guidance.

I convey my gratitude to Dr. Tatwadarshi Nagarhalli, Head of Department, and Mrs. Sejal D'Mello, Deputy HoD for their motivation and providing various facilities, which helped us greatly in the whole process of this project. I am also grateful to all other teaching and non-teaching staff members of the Artificial Intelligence and Data Science Department for directly or indirectly helping us with the completion of projects and the resources provided.

Pratik Sanjay Avhad (Roll No. 01)

Priyanka Narendra Bhandari (Roll No.02)

Swarup Satish Kakade (Roll No. 15)

Date:

Abstract

The "Smart Hindi Sign Language Interpreter using NLP" project addresses a significant communication gap between the deaf and hard-of-hearing community and the general population by developing a real-time interpreter that translates Hindi Sign Language (HSL) gestures into spoken or written Hindi. Unlike traditional systems that primarily support English, this project focuses on Hindi, making it more inclusive and relevant for a large part of India's population. By utilizing Natural Language Processing (NLP) and machine learning techniques, the interpreter recognizes sign language gestures and converts them into grammatically correct and coherent Hindi sentences. The system aims to ensure real-time processing with high accuracy, offering a seamless communication tool for both the hearing-impaired and non-signing individuals. The first phase of the project involves creating a gesture recognition model capable of accurately identifying a range of hand gestures associated with the Hindi alphabet and common expressions, forming the foundation for the interpreter's translation functionality. Alongside the technical development, a user-friendly interface is designed to make the system easily accessible and intuitive for users, regardless of their level of technical knowledge. As the system evolves, future improvements will focus on enhancing gesture recognition under varying conditions, such as poor lighting or complex backgrounds, as well as expanding the system's ability to recognize more complex signs and gestures. Additionally, the project envisions integrating augmented reality features to provide an immersive learning and communication experience. A crucial aspect of the project is its ongoing interaction with the deaf community to gather feedback, ensuring the system aligns with their real-world needs. Ultimately, this project aims to create a more inclusive society by breaking communication barriers and enabling easier interaction between the hearing-impaired community and the broader society, improving accessibility and promoting social integration.

List of Abbreviations

1. NLP - Natural Language Processing
2. CNN - Convolutional Neural Network
3. ANN - Artificial Neural Network.
4. ISL – Indian Sign Language
5. CV - Computer Vision
6. HMM - Hidden Markov Model
7. KNN - K-Nearest Neighbors
8. KNNDW - K-Nearest Neighbors with Distance Weighting
9. ML - Machine Learning
10. OTSU - Otsu's Thresholding Algorithm
11. PCA - Principal Component Analysis
12. RAM - Random Access Memory
13. SSD - Solid State Drive
14. RGB - Red, Green, Blue (color model)
15. LTK - Natural Language Toolkit
16. OpenCV - Open Source Computer Vision Library
17. EBMS - Elastic Bunch Graph Matching System
18. GPU - Graphics Processing Unit
19. HCI - Human-Computer Interaction

Table of Contents

CHAPTER NO. 1
INTRODUCTION

1.1 Introduction

In today's world, communication is a fundamental human right, but for millions of people with hearing and speech impairments, this basic right is often hindered. Hindi Sign Language (HSL) serves as a vital medium of communication for a significant part of India's population. Unfortunately, the scarcity of proficient human interpreters and resources has left many individuals unable to fully engage in society, affecting their access to education, employment, healthcare, and daily social interactions.

This is where technology can play a transformative role. The development of a **Smart Hindi Sign Language Interpreter** powered by Natural Language Processing (NLP) offers a groundbreaking solution. This system can bridge the communication gap by converting sign language gestures into comprehensible spoken or written Hindi in real time, enhancing the inclusivity of public services and daily interactions.

The integration of Natural Language Processing (NLP) with machine learning and computer vision provides a powerful approach for developing a smart Hindi Sign Language interpreter. This technology can decode hand gestures, recognize patterns in sign language, and convert them into text or speech, enabling more accessible communication.

1.2 Problem Statements & Objectives

Problem Statement:

The lack of widespread availability and accessibility of interpreters for Hindi Sign Language has led to communication barriers between the hearing-impaired community and the larger society. This gap limits opportunities for education, employment, and social inclusion for those reliant on sign language. A smart solution that can automate the interpretation of Hindi Sign Language is needed to address these challenges.

Objectives:

- To develop a smart interpreter capable of converting Hindi Sign Language gestures into spoken or written Hindi using NLP techniques.
- To ensure the system is user-friendly, accurate, and capable of real-time processing.
- To leverage machine learning models to enhance the system's ability to recognize complex signs and gestures.
- To create a bridge between the hearing-impaired and non-signing individuals, thus promoting inclusivity and accessibility.

1.3 Scope

The Smart Hindi Sign Language Interpreter project is intended to fill the gap in communication between hearing-impaired people and the Hindi-speaking public by interpreting Indian Sign Language (ISL) gestures into oral or written form using real-time video input. The system will be able to recognize hand signs via a webcam and translate them into meaningful Hindi language output.

In order to facilitate effective communication, Natural Language Processing (NLP) methods will be incorporated to convert the identified signs into grammatically accurate and contextually appropriate Hindi words or phrases. This will be done in a way that the resulting output is semantically rich and understandable.

The interpreter will be made user-friendly and accessible on various devices, such as mobile phones, tablets, and computers, so it can be easily used. User customization functions will enable users to adapt the application according to their requirements, and an in-built feedback system will be added in order to constantly enhance recognition accuracy and usage experience.

The project is also intended to facilitate educational environments, making easier learning and instruction of Hindi Sign Language possible. Future improvements will involve further advancing gesture recognition and NLP algorithms in collaboration with linguistics experts and artificial intelligence experts, ultimately enhancing the overall effectiveness and influence of the system on society.

CHAPTER NO. 2
LITERATURE SURVEY

2.1 Survey of Existing System

1) From [1] , Gesture Recognition Techniques for Sign Language Interpreter Systems

This encompasses the development of sign language interpretation systems, bridging the barriers the gestures erect, turning them into text or speech, and therefore becoming invaluable in communication to the hearing-impaired. The initiation process is typically initiated by acquiring image or video data using cameras or other media devices. This is precisely the first step of any system, and the quality of captured gestures influences the whole performance. Most systems use RGB or depth-sensing cameras, for example Kinect to capture the hand and finger movements in high resolution. It is the cornerstone for successful system execution based on good data quality.

Now, once the data for the image or video is collected, pre-processing it is done in order to prepare the frames for gesture recognition. PCA and the OTSU algorithm are primarily used here. It takes away the high dimensions of the image by representing the image in a lower space with preserved key features and removing unnecessary details. This way, the compression will make the comparison of gestures more efficient. Instead of this, the OTSU algorithm transforms the RGB image into its binary format whereby black will represent the background and white represents the hand gesture. The clear differentiation between the foreground and the background makes it possible for a more accurate extraction of the gesture to identify and isolate the gesture for further processing.[3]

The process of gesture recognition is achieved through comparison of the preprocessed image to a database of known gestures. Machine learning models such as CNNs are utilized since they can effectively map the complex input images to corresponding gestures with significant accuracy levels. PCA aids in visualizing the sequence of gestures as points in the lower-dimensional space. This helps in an efficient pattern recognition of the gesture. A similar approach adopted by the OTSU algorithm is a binary image, which filters the background data, allowing it to focus on the gesture and simplifying the recognition process. All these help pin down the intended sign language gesture for the system.[2]

The last part of the system is translating the perceived gesture in text or speech. In the event that the input matches a stored gesture, the system will produce an appropriate output. Despite

the advances made in gesture recognition, problems like differences in signing style, environmental considerations, and computational requirements still exist. To overcome these is the key for further progress of more dependable sign language interpreter systems.

2) According to [5] , Exploring Techniques in Hand Gesture Recognition for Sign Language Interpretation

Presently, hand gesture recognition research has focused on approaches to the system that could enhance its use in applications of sign language interpreters among others. The process entails four stages which include: data acquisition, preprocessing, feature extraction, and gesture classification. This review provides a general overview of the techniques and approaches used at each stage in recognition of hand gestures.

Data acquisition lays the base for hand gesture recognition. Among the two most popular methods used are sensory-based and vision-based approaches. Sensory-based methods make use of electromechanical devices such as gloves, which capture a detailed hand configuration besides positions. However, such devices are usually very expensive, less user-friendly, and not very practical for worldwide use. This ability to handle such variations is critical to enhancing the accuracy of recognition in vision-based systems.

To be used in classification, it must be preprocessed and from the given data, features to be extracted. For a vision-based approach, this will include background subtraction, color detection, and in the case of actual images, some amount of filter application. The work in [5] detects a hand by combining threshold-based color detection with the use of background subtraction such that the hand is separated from the other elements in the captured image. Also, the Adaboost face detector is used to distinguish between hands and faces because hands as well as faces tend to have similar types of skin tones. Image filters such as Gaussian blur are applied to remove noise but increase the quality of the picture so the significant features needed to successfully identify a gesture may be more easily separated.[6]

Gesture classification is the final step of hand gesture recognition, depending upon various machine learning algorithms for the identification and categorization of gestures. HMM would be very effective for modeling dynamic gestures through the capturing of their temporal characteristics, whereas Naïve Bayes Classifiers are the best for static gestures classification with geometric invariants extracted from the segmented images across a variety of skin tones. Another method is the K Nearest Neighbor (KNN) algorithm, strengthened by a distance-weighted method called KNNDW, which provides good performance for the recognition of a number of gestures. Another promising method is through Convolutional Neural Networks

(CNNs) which isolate the hand with a skin model and a binary thresholding process prior to training. Recent work done by Hsien-I Lin and colleagues demonstrates CNN for the performance of gesture recognition tasks with an accuracy of up to 95% using seven hand gestures. This indicates that the increased performance of a gesture recognition system through the use of deep learning techniques is possible in sign language interpretation.[7].

Hand gesture recognition has seen tremendous growth over the years, primarily due to vision-based methods and machine learning approaches. With such methods using algorithms like HMM, Naïve Bayes, KNN, and CNN, the recognition accuracy and efficiency have improved within gesture recognition systems. However, things are still problematic in dealing with variations of hand appearances as well as environmental conditions. On-going research in these aspects will continue to produce more robust and effective sign language interpretation systems, bringing about wider accessibility and social benefits.[9]

3) From the article mentioned in [11] ,Approaches for Hand Gesture Recognition

Hand gesture recognition is a very important research area to which many approaches contribute. In this chapter, the literature is divided into three main categories: image processing/statistical modeling-based recognition, classic machine learning-based recognition, and deep learning-based recognition.

One of the first studies by Triesch and Malsburg (1996) described a highly accurate American Sign Language recognition system with an extremely low error rate, even in complex backgrounds. In this study, they assumed hand images as input, thus obviating the need for hand segmentation, and employed Gabor filters for feature extraction while simulating receptive fields of the visual cortex. Utilizing the technique EBGM (Elastic Bunch Graph Matching), their method achieved 86.2% accuracy under complex conditions and, on average, 91% using a dataset consisting of 657 images from 24 subjects. [12]

The use of Hidden Markov Models for gesture recognition has popularized the problem setting of gesture recognition as a problem of action recognition. How-ever, newer classic approaches linked with techniques from machine learning and deep learning reveal bright perspectives in enhancing hand gesture recognition systems' accuracy and robustness.

4) As per [13], recent sign language processing efforts have included varied tasks like sign detection, gloss recognition, translation, and generation, with growing interest in Indian Sign Language (ISL). Though there has been improvement in ASL and other sign languages, ISL continues to lack annotated and publicly available datasets. Some of the available resources are

the INCLUDE dataset and ISL-CSLRT, which provide video data for static and continuous ISL gestures respectively. Methods like RGB and depth modalities have been investigated for gesture recognition, though restricted access to datasets is still a hindrance to large-scale NLP-based model training. As mentioned in [14], researchers have employed a vast array of methodologies for ISL gesture recognition, ranging from traditional approaches such as Eigen value-based Euclidean classifiers, ANN, and Otsu segmentation, to more recent hybrid approaches fusing KNN with histogram-based features and PCA with SVM. The addition of CNN and LSTM models has further improved recognition accuracy for dynamic gestures, with examples ranging from SURF feature extraction to enhance rotation invariance and decrease processing time. In [15], the sensor-based and vision-based methods comparison underscored the cost-effectiveness and practicality of computer vision approaches, in addition to illustrating the efficacy of deep learning models like CNNs, RNNs, and GANs. ISL recognition has utilized methods such as YCbCr segmentation, Wavelet Packet Decomposition, and 3D-CNNs for dynamic signs. Research integrating these models with NLP and GAN architectures has demonstrated high performance, pointing to an increasing trend towards end-to-end ISL interpretation systems with the ability to recognize static alphabets as well as full sentences.

Paper	**System**	**Implementation**	**Technology Used**
Lin et al. CNN approach	Gesture Recognition using Convolutional Neural Networks	Isolated hand using a skin model and binary thresholding before feeding into CNN for training and prediction	Convolutional Neural Networks (CNN)
HMM-based Gesture Recognition (various sources)	Hand Gesture Recognition with HMM	Models gesture recognition as an action recognition problem, capturing temporal characteristics	Hidden Markov Models
Triesch and Malsburg (1996)	ASL Hand Gesture Recognition System	Eliminated hand segmentation; used Gabor filters for feature extraction and Elastic	Gabor Filters, EBGM

		Bunch Graph Matching (EBGM)	
Proposed ISL Recognition Method	ISL Gloss Recognition using Surf And Deep Learning	Custom dataset with SURF feature extraction and CNN-LSTM-based gesture classification.	SURF Features, CNN, LSTM

Table 2.1. Analysis of existing systems

2.2 Limitation In Existing System or Research Gap

While hand gesture recognition has witnessed significant progress, current systems have a number of challenges. Classical image processing methods tend to perform poorly under less-than-ideal conditions such as low lighting, cluttered backgrounds, or occlusions, and accuracy is compromised. They depend significantly on clean segmentation of hands, which in real-world settings is hard to obtain.

Machine learning algorithms, particularly HMM-based ones, find it difficult to identify dynamic or rapid gestures and have limited generalization between various users having different hand sizes, shapes, and skin tones. Although deep learning techniques such as CNNs provide improved performance, they are needed with big, well-tagged datasets—unavailable in plenty, particularly for Indian Sign Language.

Additionally, the absence of standardized datasets hinders the comparison of various models on an equal basis. These issues point to the necessity of strong, adaptive, and inclusive gesture recognition systems that can operate in various environments and user conditions. Bridging these gaps is important for creating a practical and accurate sign language interpreter with integrated meaningful NLP translation.

2.3 Mini Project Contribution

The Smart Hindi Sign Language Interpreter project serves a critical function in improving communication between the hearing-impaired and Hindi-speaking people. Since English is not

spoken throughout India in all areas, the decision to translate signs into Hindi makes the system more comprehensive and effective for a larger group of people.

The system employs cutting-edge technologies such as Convolutional Neural Networks (CNN) for sign recognition and Natural Language Processing (NLP) for translating recognized signs into spoken or written Hindi. This not only enhances the speed and accuracy of interpretation but also invites more research in regional language-based sign recognition systems. It aids the development of a multilingual Hindi Sign Language (HSL) dataset, which can be helpful for future model training and comparison.

The intuitive interface supports users with different degrees of technical skills and can be customized in accordance with individual requirements. Moreover, the system is potentially used as a teaching aid for HSL instruction in schools and educational institutions. On a social level, the project impacts positively by spreading awareness on communication issues among the hearing-impaired and supports inclusivity and accessibility in day-to-day interactions.

CHAPTER NO. 3
PROPOSED SYSTEM

The Smart Hindi Sign Language Interpreter should be a system to automatically identify sign language gestures and then translate those into text so that initially, it can give outputs in English and finally, in Hindi. Then, it will allow the user to have Hindi outputs translated through text-to-speech functionality in the final stage of development. The methodology involves the use of computer vision, CNNs, and NLP in order to interpret sign language and aid anyone who is hearing and speech impaired in communication.

Some of the important features of the proposed system are:

- Sign Language to Text: In this program, we recognize and translate hand gestures of alphabets (A–Z) and numbers (0–9) into text.
- Translation to Hindi: The recognized English output is translated to Hindi using Translator API.
- Text-to-Speech: The final Hindi output is converted into speech for audio-based communication.
- User-Friendly Interface: The intuitive interface (built using Flask and HTML) makes the user interaction direct and simple.

Methodology

1. Objective:

   The system collects webcam data, scales and aligns the hand gestures, and classifies them using a CNN model to generate output in text form. It recognizes English alphabets and digits, translates them to Hindi using an API, and converts that to speech output in real time.

2. Data Collection (dataset_keypoint_generation.py):
   i. The system captures hand gesture data through the webcam. HandDetector identifies and isolates the hand region.
   ii. Hand Detection: A pre-trained model locates the hand region, extracts a bounding box, and crops it to 300x300 pixels.

iii. Data Preprocessing: Each hand image is resized and placed on a blank 300x300 white canvas for uniform training input.
iv. Saving the Data: The processed images are stored in folders labeled A-Z and 0-9 for classification.

3. Real-Time Gesture Recognition (isl_detection.py):
   i. Hand Detection: During real-time prediction, the HandDetector identifies and isolates the hand from webcam feed.
   ii. Image Preprocessing: The cropped hand is resized and aligned on a blank white canvas of 300x300 pixels.
   iii. Gesture Classification: A trained CNN (model.h5) classifies the gesture into an alphabet or digit.
   iv. Translation and Display: The recognized text is displayed in English, translated to Hindi using an API, and finally spoken out using a text-to-speech module.

4. Translation to Hindi
   i. Translation Module: The recognized English text is translated into Hindi using a language translation API (e.g., Google Translate API or similar).
   ii. Dynamic Conversion: This translation occurs instantly after gesture recognition to ensure seamless communication.

5. Text-to-Speech Conversion
   i. Speech Synthesis: The Hindi output is converted into spoken audio using a text-to-speech (TTS) engine (e.g., gTTS or pyttsx3).
   ii. Audio Playback: The user hears the translated output, supporting effective verbal communication.

6. User-Friendly Interface (Frontend)
   i. Dashboard Design: The web interface features a clean, modern layout with clearly defined sections Detect, Learn, Practice, and About for intuitive navigation
   ii. Real-Time Display: Recognized characters, Hindi translation, and audio playback are all presented clearly on the interface for immediate feedback.

3.1 Architecture/Framework /Block Diagram

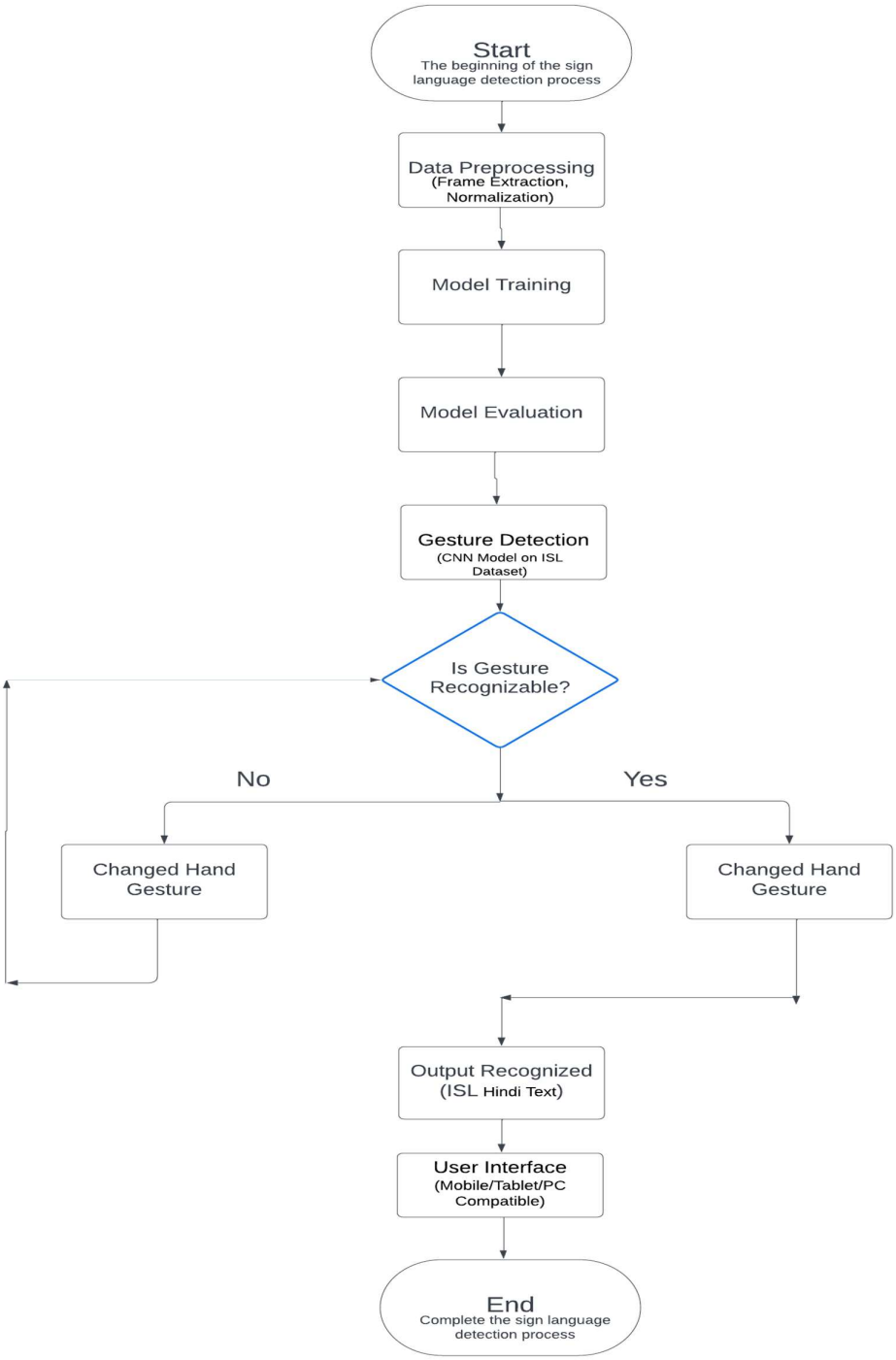


Fig 3.1 Block Diagram

3.2 Algorithm And Process Design

Initialize Resources:

i. Initialize Camera: Open the webcam for video capture with cv2.VideoCapture().
ii. Initialize HandDetector: Use HandDetector from cvzone for detection up to 2 hands.
iii. Classifier Initialization (only for testing): Load the trained hand gesture classifier model using Classifier() based on ISL alphabet dataset.

1. Capture Video Frames:

i. Capture Frame: Grab a frame from webcam continuously with cap.read(). If it captures successfully, proceed. If failed, show an error message and move on.
ii. Copy (for testing): Maintain a copy of the frame for later output display
iii. Hand Detection:

a. Detect Hands: Apply the hand detector on hands in the current frame.
b. If Hands Detected: Verify if at least one hand was detected. If hands are detected, extract the coordinates of the bounding box of the first hand: x = left border, y = top border, w = width, h = height.

2. Crop and Resize Hand Region:

i. Create a White Blank Image: Create a blank white canvas of size 300x300 pixels, imgWhite
ii. Crop Hand Region: Extract the bounding box coordinates enclosing the frame region containing the hand with an appropriate offset.
iii. Aspect Ratio Handling: Check whether the aspect ratio (height-to-width) of the cropped hand image is greater than the width
iv. If height > width, resize the image based on height and paste centrally on the blank white canvas.
v. If width > height, resize the image based on width and paste centrally on the blank white canvas.

3. Show Cropped and Resized Image:

i. Show Cropped Image: Display the cropped image of the hand
ii. Show Processed Image: Display the final 300x300 image (imgWhite) containing the resized hand.

4. Save the Image (Data Collection Phase Only):

i. Save Processed Image: During data gathering, once the 's' key is typed, save the processed image (imgWhite) in the appropriate folder with a unique filename based on the timestamp.
ii. Increment the Counter: Increment the counter value for each saved image.
iii. Exit the Program: Once a 'q' key is received, the program must quit and terminate the webcam.

5. Classify Gesture and Form Words (Real-Time Testing Phase Only):

i. Make a Prediction: Pass the processed hand image (300x300) to the loaded classifier model (classifier.getPrediction()) to recognize ISL alphabets.
ii. Buffer Alphabet: Store the recognized alphabet in a temporary buffer.
iii. Word Formation: Continuously monitor the buffer for a sequence of alphabets. When a word boundary (e.g., pause or specific gesture) is detected, combine the alphabets into a word.
iv. NLP Translation: Translate the formed word from ISL alphabets to Hindi text or speech using NLP.
v. Display Prediction: Draw a rectangle over the hand and display the predicted Hindi word (e.g., "नमस्ते" for "namaste") over the video output.

6. Display Final Output (Real-Time Testing Phase Only):

i. Show Final Output: Display the original frame with a bounding box surrounding the hand and the predicted Hindi word overlaid.
ii. Exit the Program: When the 'q' key is pressed, quit the program and release the webcam.

3.3 Details Of Hardware And Software

Hardware Details:

1. **Edition**: Windows 11 Home or equivalent under Linux, such as Ubuntu 20.04.
2. **Processor**: Intel Core i5-11400 or better, operating at 2.60 GHz.
3. **RAM**: At least 8 GB, 16 GB or more will be desired for better performance
4. **Storage**: 512 GB of SSD for fast reads and writes of large image datasets.
5. **Camera**: A webcam with at least 720p resolution, although a 1080p resolution camera is used.
6. **GPU** (Optional but Recommended): NVIDIA GTX 1650 or more for training and inference in real-time via deep learning models.
7. **System Type**: It must support a 64-bit architecture so that it must be an x64-based processor architecture to train the model with proper speed.

Software Details:

1. **Programming Language**: The image-processing, hand gesture recognition, and natural language processing would be implemented using Python 3.10.10.
2. **Image Processing Library**: OpenCV to handle image capture, processing and resizing of hand gesture images.
3. **Hand Tracking Library**:
   i. CVZone- Handle the real-time hand tracking and gesture detection.
   ii. MediaPipe (optional) - an alternative library for more accurate gesture recognition.
4. **Numerical Computation Library**: NumPy for matrix manipulations and efficient numerical operations on the image data.
5. **Machine Learning Library**: TensorFlow/Keras, for development, training, and deployment of the deep learning model for interpreting hand gestures.
6. **Classifier Library**: cvzone.ClassificationModule, to utilize the trained model to classify the hand gestures in real time.
7. **Natural Language Processing**: NLP libraries like NLTK or spaCy, for processing of the language and acquiring the text based on the interpretations of hand gestures made during later stages.
8. **Development Environment**:
   i. Anaconda, for managing Python environments and its dependencies.

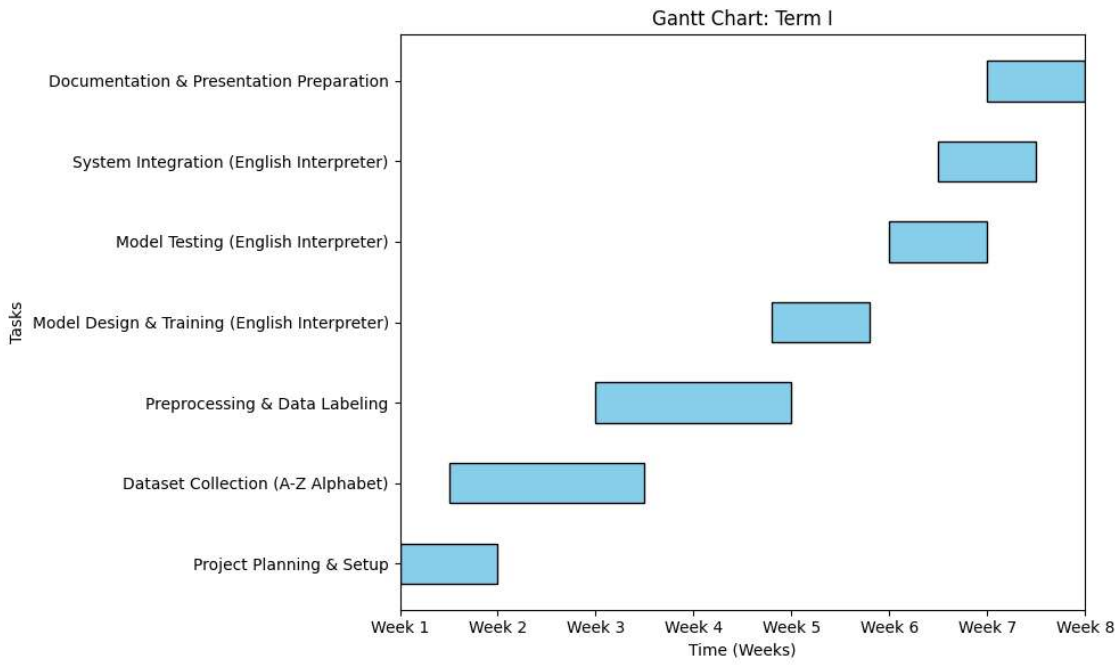
ii. Jupyter Notebook: When prototyping and testing the code during the development phase, the notebook used is typically for iteration purposes.

iii. Integrated Development Environment (IDE): For the writing, debugging, and managing the project's code base, you would primarily use either PyCharm or Visual Studio Code

9. **Version contro**l : Git for version control purposes, if it is a team development
10. **Deep Learning Hardware Acceleration Optional**: If on GPU, one has to employ the CUDA and cuDNN libraries and other APIs to enhance acceleration specifically for deep learning-oriented tasks like model training and inference.
11. **Frontend Development**:
    i. Framework: React.js (with functional components and hooks).
    ii. Bundler: Vite, for faster and optimized development experience.
    iii. Styling: Tailwind CSS or CSS Modules for responsive and modern UI design.
    iv. Routing: React Router for navigation between pages.
    v. API Handling: Axios or Fetch API to communicate with the Flask backend.

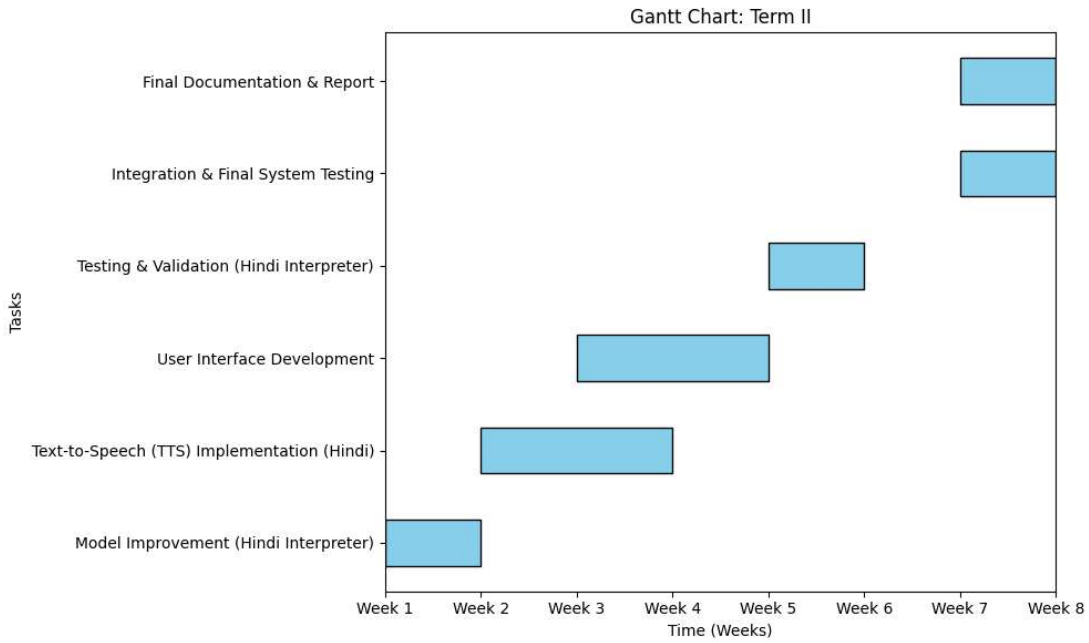
CHAPTER NO. 4
IMPLEMENTATION PLAN

4.1 Gantt Chart (Term I and Term II)



Gantt Chart: Term I (English Interpreter Development)

The Term I Gantt chart outlines the foundational phase of developing an English Sign Language Interpreter system. It starts in Week 1 with project planning and setup, which includes task allocation, defining objectives, and selecting appropriate tools and frameworks. From Week 2 to Week 4, the focus shifts to dataset collection of A–Z sign alphabets and preprocessing & data labeling, where raw gesture data is cleaned and annotated for model training. Week 5 involves the design and training of the initial machine learning model tailored to the English sign inputs. This is followed by model testing in Week 6, where accuracy, speed, and responsiveness are evaluated. In Week 7, the trained model is integrated into a system that includes frontend and backend components. The final Week 8 is allocated for documentation and presentation preparation, where results, challenges, and methodology are compiled. This term ensures a complete pipeline—from planning, data handling, training, to partial deployment—making the English interpreter system functional and ready for improvements.



Gantt Chart: Term II (Hindi Interpreter Development)

The Term II Gantt chart focuses on the enhancement and extension of the system to support the Hindi Sign Language Interpreter with added features like Text-to-Speech (TTS). Starting from Week 1, efforts are made toward improving the existing model to adapt to Hindi gestures, ensuring better accuracy for Indian regional contexts. From Week 2 to Week 4, the core TTS module is implemented to convert recognized gestures into spoken Hindi, significantly enhancing accessibility for visually impaired users. Simultaneously, user interface development is undertaken to provide a clean, farmer-friendly, and multilingual UI. Week 5 is dedicated to testing and validation, ensuring the Hindi interpreter system functions accurately and efficiently under various real-world conditions. Week 7 includes final integration and system testing, where all components—model, TTS, and UI—are brought together and rigorously tested. Week 8 concludes with final documentation and reporting, where project insights, results, future scope, and user feedback are documented. This term finalizes the bilingual interpreter system, making it robust, interactive, and ready for deployment in real-time assistive technologies.

5.2 Result , Testing And Analysis

1. Objectives

The primary objective of the project is to develop an intelligent interpreter that translates Hindi text and speech into sign language, facilitating effective communication between the hearing and deaf communities. By focusing on inclusivity, the project aims to empower users by enhancing their ability to communicate in diverse settings, including education, healthcare, and social interactions.

2. Methodology

The project employs a multi-phase approach:

i. Data Collection: A comprehensive dataset of sign language gestures corresponding to the Hindi alphabet is gathered. This dataset serves as the foundation for training the machine learning models.
ii. Model Development: Initial development focuses on creating a robust model for interpreting English, which will be adapted for Hindi in later phases. Techniques in natural language processing and machine learning are utilized to ensure high accuracy and reliability.
iii. User Interface Design: A user-friendly interface is developed to ensure accessibility for all users, allowing for easy interaction with the interpreter.
iv. Testing and Feedback: Engaging with the deaf community for user testing and feedback is crucial for refining the model and improving its usability.

3. Strengths

i. Inclusivity: The project addresses a significant gap in communication for the deaf and hard-of-hearing communities, promoting inclusivity.
ii. Technology-Driven: Utilizes advanced NLP and machine learning techniques, ensuring the interpreter is at the forefront of technology.
iii. User-Centric Design: Involvement of the deaf community in the testing phase ensures the tool is tailored to user needs, enhancing acceptance and usability.

4. Weaknesses

i. Data Collection Challenges: Gathering a diverse and representative dataset can be difficult, especially in regions with different sign languages or dialects.
ii. Real-time Processing Limitations: Achieving real-time interpretation may be technically challenging, requiring significant computational resources and optimization.
iii. Dependency on Technology: Users with limited access to technology may find it difficult to benefit from the interpreter, potentially limiting its reach.

5. Opportunities

i. Expanding Language Support: There is potential to expand the interpreter to support additional languages, catering to a wider audience and enhancing inclusivity.
ii. Partnerships: Collaborations with educational institutions, NGOs, and government organizations could facilitate deployment and enhance the project's impact.
iii. Integration with AR/VR: Exploring augmented reality or virtual reality applications can create immersive experiences that enhance learning and communication.

6. Threats

i. Competition: There may be existing tools or future developments that compete with this interpreter, which could limit its market share and relevance.
ii. Technological Changes: Rapid advancements in technology may necessitate constant updates and adaptations to the interpreter, requiring ongoing investment.
iii. User Adoption: Gaining acceptance from the target audience may be challenging, especially if users are accustomed to traditional communication methods.

7. Potential Impact

The "Smart Hindi Sign Language Interpreter" has the potential to significantly impact communication accessibility for the deaf and hard-of-hearing communities. By facilitating better interactions in education and social settings, it promotes inclusion and understanding. Furthermore, the project may serve as a model for future initiatives aimed at bridging communication gaps in other languages and cultures.

CHAPTER NO. 6
CONCLUSION AND FUTURE WORK

This project marks a significant step forward in bridging the communication gap for the hearing-impaired community, particularly in India. Recognizing that English is not universally accessible across the country, the system incorporates Hindi, the most widely spoken language, to create a more local and inclusive solution. By translating sign language into Hindi, it empowers individuals who rely on sign language, making communication with the broader society more fluid and accessible.

The backend is built using Python, with OpenCV and CVZone handling gesture recognition, NumPy for efficient data handling, and Natural Language Processing (NLP) libraries like NLTK and spaCy to convert recognized gestures into coherent Hindi phrases. Development tools such as Anaconda for environment management and Jupyter Notebook for prototyping helped streamline the process and enhance productivity.

In addition to the core logic, a user-friendly frontend has been developed using React and Vite, providing a fast, modern, and responsive interface. This enhances accessibility for users and makes the system more approachable, especially for those with limited technical background. The intuitive design ensures a smooth interaction between users and the interpreter, significantly improving the overall user experience.

While the system performs well, challenges such as varying lighting conditions, complex backgrounds, and the need for more diverse datasets remain areas for future improvement. Addressing these limitations will be essential for ensuring robust performance across different users and environments.

To conclude, this project is not only a technological achievement but also a socially impactful solution. It fosters accessibility and inclusivity by enabling smoother communication for sign language users. With continued development, the system holds great promise to improve the quality of life for the deaf and hard-of-hearing community, helping them connect more seamlessly with the world around them.

Future Work

1. Real-time Sign Language Interpretation via Live Video: Develop and refine real-time sign language recognition using live camera input. This feature would enable spontaneous and natural conversations in settings like classrooms, hospitals, and public services.

2.Adaptive and Personalized User Experience: Expand the user interface with intelligent customization options—such as auto-adjusting gesture sensitivity, dark/light modes, and tailored learning paths based on user behavior and preferences.

3. Feedback-Driven Model Improvement: Implement a built-in feedback system where users can rate interpretations, report errors, or suggest corrections. These insights can be used to continuously fine-tune the model for better accuracy and relevance.

4. Augmented Reality (AR) Integration: Integrate AR capabilities to overlay sign language or text translations in the user's environment through mobile devices or smart glasses, enhancing learning and real-time communication in immersive ways.

5.Cross-Platform Integration (e.g., Zoom, MS Teams): Enable plugin or API support for popular video conferencing tools, making sign language interpretation accessible in online meetings, classrooms, and webinars.

6. Community Collaboration Hub: Create a platform where users especially from the deaf and hard-of-hearing communities can contribute new signs, regional variations, and training data, turning the interpreter into a community-driven resource.

By pursuing these avenues, we can create a powerful tool that not only bridges communication gaps but also enriches the lives of individuals within the deaf and hard-of-hearing communities.

REFERENCE

[1] M. K. Ahuja and A. D. Singh, "Hand Gesture Recognition Using PCA," International Journal of Computer Science and Emerging Technologies (IJCSET), vol. 6, no. 7, pp. 385-390, July 2015.

[2] Z. Yang, Y. Li, W. Chen, and Y. Zheng, "Dynamic Hand Gesture Recognition Using Hidden Markov Models," in Proceedings of the 7th International Conference on Computer Science & Education (ICCSE), Melbourne, Australia, 2012, pp. 360-365.

[3] P. R. V. Chowdary, M. N. Babu, T. V. Subbareddy, B. M. Reddy, and V. Elamaran, "Image Processing Algorithms for Gesture Recognition using MATLAB," in IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), Ramanathapuram, India, 2014, pp. 902-906.

[4] E. R. Selvamani, K. S. Kanimozhi, V. Rao, and A. Kannan, "Intelligent System for Gesture Recognition," in Proceedings of the International Conference on Intelligent Systems and Control, Coimbatore, India, 2013, pp. 150-154.

[5] T. Yang and Y. Xu, "Hidden Markov Model for Gesture Recognition," CMU-RI-TR-94-10, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, May 1994.

[6] A. Ianai, T. Müller, M. E. Foster, and A. Knoll, "A Naïve Bayes Approach for Gesture Recognition," Dept. of Informatics VI, Robotics and Embedded Systems, Boltzmannstr. 3, DE-85748 Garching, Germany.

[7] X. Chen, Y. Li, R. Hu, X. Zhang, and X. Chen, "Hand Gesture Recognition based on Surface Electromyography using Convolutional Neural Network with Transfer Learning Method," 2020

[8] R. Rastgoo, K. Kiani, and S. Escalera, "Sign Language Recognition: A Deep Survey," Electrical and Computer Engineering Department, Semnan University, Semnan, Iran, Oct. 2020.

[9] A. Wadhawan and P. Kumar, "Sign Language Recognition Systems: A Decade Systematic Literature Review," Archives of Computational Methods in Engineering, 2019.

[10] Z. Murtaza, H. Akmal, W. Afzal, H. E. Gelani, Z. ul Abdin, and M. H. Gulzar, "Human-Computer Interaction Based on Gestural Cues Recognition/Sign Language to Text Conversion," in 2019 International Conference on Engineering and Emerging Technologies (ICEET), Lahore, Pakistan, Feb. 2019, pp. 1-6.

[11] . Gu, X. Yuan, and T. Ikenaga, "Hand Gesture Interface Based on Improved Adaptive Hand Area Detection and Contour Signature," in IEEE International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), 2012, pp. 463-468.

[12] H. Y. Lai and H. J. Lai, "Real-Time Dynamic Hand Gesture Recognition," in IEEE International Symposium on Computer Consumer Control, 2014, no. 1, pp. 658-661.

[13] A. Joshi, A. Bhat, P. S., P. Gole, S. Gupta, S. Agarwal, and A. Modi, "CISLR: Corpus for Indian Sign Language Recognition," in Proc. 2022 Conf. Empirical Methods in Natural Language Processing, Abu Dhabi, United Arab Emirates, 2022, pp. 10357–10366.

[14] S. Katoch, V. Singh, and U. S. Tiwary, "Indian Sign Language recognition system using SURF with SVM and CNN," Array, vol. 14, p. 100141, 2022.

[15] P. P. Waghmare, "Deep Learning Approach for Combined Indian Sign Language Recognition and Video Generation Model," Int. J. Intell. Syst. Appl. Eng., vol. 12, no. 4, pp. 3296–3302, 2024.

MINI PROJECT OUTCOME

Research Paper:

157 of 1,423

Notification of Paper Non-Acceptance - Fifth IEEE International Conference on Computing and Communications Technologies (ICCCT'25) External Inbox ×

Microsoft CMT <email@msr-cmt.org> Mon, Mar 24, 7:19 PM
to me

Dear Swarup Satish Kakade,

Thank you for your submission titled "3052 - Smart Hindi Sign Language Interpreter Using NLP". We appreciate the time and effort you have dedicated to your research and in preparing your submission.

After a thorough review by our esteemed reviewers, we regret to inform you that your paper has not been accepted for publication/presentation. The decision was based on various factors, including Scope, Reviewer Comments and Quality Concerns.

We sincerely value your contribution and look forward to your participation in our future events.

Best regards,
ICCCT 2025 Team

To stop receiving conference emails, you can check the 'Do not send me conference email' box from your User Profile.

Microsoft respects your privacy. To learn more, please read our Privacy Statement.

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

VNPS Certificate:


VIDYAVARDHINI'S COLLEGE OF ENGINEERING AND TECHNOLOGY
(NBA & NAAC ACCREDITED)
K.T. MARG, VASAI ROAD (W), PALGHAR - 401 202
FOUNDER PRESIDENT LATE PADMASHREE H.G.VARTAK
(AFFILIATED TO UNIVERSITY OF MUMBAI)
VNPS'25
CERTIFICATE
OF PARTICIPATION
THIS CERTIFICATE IS PROUDLY PRESENTED TO
Pratik Sanjay Avhad
in recognition of their participation in VNPS'25 held on 4th April, 2025
Branch Counsellor
Co-Convener
Convener
Principal
In Association with Student Chapters of
IEEE
NSDC
ISHRAE
IETE
CSI
VMEA
IGBC


VIDYAVARDHINI'S COLLEGE OF ENGINEERING AND TECHNOLOGY
(NBA & NAAC ACCREDITED)
K.T. MARG, VASAI ROAD (W), PALGHAR - 401 202
FOUNDER PRESIDENT LATE PADMASHREE H.G.VARTAK
(AFFILIATED TO UNIVERSITY OF MUMBAI)
VNPS'25
CERTIFICATE
OF PARTICIPATION
THIS CERTIFICATE IS PROUDLY PRESENTED TO
Swarup Satish Kakade
in recognition of their participation in VNPS'25 held on 4th April, 2025
Branch Counsellor
Co-Convener
Convener
Principal
In Association with Student Chapters of
IEEE
NSDC
ISHRAE
IETE
CSI
VMEA
IGBC