RECOGNITION OF INDIAN SIGN LANGUAGE BY USING HAND GESTURES AND SPEECH GENERATION

A PROJECT REPORT

Submitted by

GAJA LAKSHMI J (1919102041)
BHUVANA K (1919102026)
DINESH RAJ V M (1919102039)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

SONA COLLEGE OF TECHNOLOGY
(An Autonomous Institution)
SALEM-636005

ANNA UNIVERSITY: CHENNAI-600 025

MAY 2023

SONA COLLEGE OF TECHNOLOGY
(An Autonomous Institution)
SALEM-636005

ANNA UNIVERSITY: CHENNAI 600025

BONAFIDE CERTIFICATE

Certified that this project report **"RECOGNITION OF INDIAN SIGN LANGUAGE BY USING HAND GESTURES AND SPEECH GENERATION"** is the bonafide work of **"Gaja Lakshmi J (1919102041), Bhuvana K (1919102026), Dinesh Raj V M (1919102039)"** who carried out the project work under my supervision.

SIGNATURE
Dr. B. SATHIYABHAMA, B.E., M.Tech.,Ph.D

HEAD OF THE DEPARTMENT
Professor
Department of Computer Science and Engineering
Sona College of Technology,
Salem – 636005

SIGNATURE
Prof. D. Vidyabharathi

SUPERVISOR
Associate Professor
Department of Computer Science and Engineering
Sona College of Technology,
Salem – 636005

Submitted for the project viva-voce examination held on ……..

_____ _____

INTERNAL EXAMINER EXTERNAL EXAMINER


JEPPIAAR
ENGINEERING COLLEGE
(A Christian Minority Institution)
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600119
8th IEEE SPONSORED INTERNATIONAL CONFERENCE ON SCIENCE, TECHNOLOGY, ENGINEERING AND MATHEMATICS
ICONSTEM 2023
"Smart Systems and Innovative Technology for Industry 5.0"
CERTIFICATE OF PARTICIPATION
Paper ID: ICON23T1239
This is to certify that Gaja Lakshmi.J of
Sona College Of Technology
has actively participated and presented a research paper in ICONSTEM 2023 titled
" Real Time Indian Sign Language Recognition using Hand Gesture and Text/Voice
Generation "
in the 8th IEEE Sponsored International Conference on Science, Technology, Engineering and Mathematics (ICONSTEM 2023) organized by Jeppiaar Engineering College on 6th & 7th April, 2023.
Dr. J. Jebastine, M.E, Ph.D
SB Counsellor, IEEE SB JEC
Convenor, ICONSTEM 2023
Dr. Shaleesha A. Stanley, M.Sc, M.Tech, Ph.D, D.Sc
Dean - Academics & HOD-BioTech
Jeppiaar Engineering College
Dr. J. Francis Xavier, M.Tech, Ph.D
Principal
Jeppiaar Engineering College
Dr. M. Regeena J Murali, B.Tech, M.B.A, Ph.D
Founder & Chancellor, Jeppiaar University
Chairman and Managing Director
Jeppiaar Group of Institutions


JEPPIAAR
ENGINEERING COLLEGE
(A Christian Minority Institution)
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600119
8th IEEE SPONSORED INTERNATIONAL CONFERENCE ON SCIENCE, TECHNOLOGY, ENGINEERING AND MATHEMATICS
ICONSTEM 2023
"Smart Systems and Innovative Technology for Industry 5.0"
CERTIFICATE OF PARTICIPATION
Paper ID: ICON23T1239
This is to certify that Dinesh Raj.V.M of
Sona College Of Technology
has actively participated and presented a research paper in ICONSTEM 2023 titled
" Real Time Indian Sign Language Recognition using Hand Gesture and Text/Voice
Generation "
in the 8th IEEE Sponsored International Conference on Science, Technology, Engineering and Mathematics (ICONSTEM 2023) organized by Jeppiaar Engineering College on 6th & 7th April, 2023.
Dr. J. Jebastine, M.E, Ph.D
SB Counsellor, IEEE SB JEC
Convenor, ICONSTEM 2023
Dr. Shaleesha A. Stanley, M.Sc, M.Tech, Ph.D, D.Sc
Dean - Academics & HOD-BioTech
Jeppiaar Engineering College
Dr. J. Francis Xavier, M.Tech, Ph.D
Principal
Jeppiaar Engineering College
Dr. M. Regeena J Murali, B.Tech, M.B.A, Ph.D
Founder & Chancellor, Jeppiaar University
Chairman and Managing Director
Jeppiaar Group of Institutions

JEPPIAAR ENGINEERING COLLEGE

(A Christian Minority Institution)

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600119

8th IEEE SPONSORED INTERNATIONAL CONFERENCE ON SCIENCE, TECHNOLOGY, ENGINEERING AND MATHEMATICS

ICONSTEM 2023

"Smart Systems and Innovative Technology for Industry 5.0"

CERTIFICATE OF PARTICIPATION

Paper ID: ICON23T1239

This is to certify that Bhuvana.K of Sona College Of Technology has actively participated and presented a research paper in **ICONSTEM 2023** titled "Real Time Indian Sign Language Recognition using Hand Gesture and Text/Voice Generation"

in the 8th IEEE Sponsored International Conference on Science, Technology, Engineering and Mathematics (ICONSTEM 2023) organized by Jeppiaar Engineering College on 6th & 7th April, 2023.

Dr. J. Jebastine, M.E, Ph.D
SB Counsellor, IEEE SB JEC
Convenor, ICONSTEM 2023

Dr. Shaleesha A. Stanley, M.Sc, M.Tech, Ph.D, D.Sc
Dean - Academics & HOD-BioTech
Jeppiaar Engineering College

Dr. J. Francis Xavier, M.Tech, Ph.D
Principal
Jeppiaar Engineering College

Dr. M. Regeena J Murali, B.Tech, M.B.A, Ph.D
Founder & Chancellor, Jeppiaar University
Chairman and Managing Director
Jeppiaar Group of Institutions

PAPER ID: ICON23T1239

8 th IEEE International Conference on Science, Technology, Engineering and Mathematics (ICONSTEM 2023)

Publication Partner: IEEE *Chennai, Tamil Nadu, INDIA*

Real Time Indian Sign Language Recognition using Hand Gesture and Text/Voice Generation

Vidyabharathi.D, Gaja Lakshmi.J, Bhuvana.K, Dinesh Raj.V.M

Professor, Department of CSE, Sona College of Technology,Salem, India

Department of CSE,Sona College of Technology,Salem, India

Abstract—Communication with a person with hearing impairment is always a great challenge. Disabled people with hearing loss use sign language as a means of communicating with others. Hand gestures are one of the ways that sign language users can communicate nonverbally. Many people around the world have developed different sign language to communicate in their native language. In this paper, we present a proposed system for recognizing Indian Sign Language (ISL) based on hand gestures and generate the speech output for the recognized text. The purpose of this work is to develop a model that can recognize hand gestures in real time using computer vision and then the model is trained to generate the appropriate character, words, or sentences for the recognized sign. In order to enhance communication between the person with hearing impairment and the blind, this technology also offers voice output for the generated text. Additionally, this system offers the text to sign language generation paradigm, which enables two-way communication without the use of a translator.

Keywords- Indian Sign Language (ISL), Hand gestures, Computer Vision, Convolutional Neural Networks (CNN), Voice Output, Two-way communication

ACKNOWLEDGMENT

First and foremost, I would like to express my gratefulness to our honorable Chairman **Sri. C Valliappa** our Vice Chairman **Sri. Chocka Valliappa** and **Sri Thyagu Valliappa** and the management of Sona College of Technology for bring me constant encouragement throughout this course

My Sincere thanks goes to **Dr. S. R. R. Senthil Kumar**, Principal. Sona College of Technology, who has motivated me in my entire endeavors. I wholeheartedly thank Dr**. B. Sathiyabhama**, Professor and Head Department of Computer Science and Engineering, Sona College of Technology Salem for giving constant encouragement and rendering all kinds of support throughout the course.

I use this opportunity to express my deepest gratitude and special thanks to my project guide **Prof.D.Vidyabharathi, Associate Professor** Department of Computer Science and Engineering, Sona College of Technology.

Special thanks go to my class counsellor Prof. **S. Theetchenya, Assistant Professor,** Department of Computer Science and Engineering, Sona College of Technology

I would like to thank my parents and all my friends from the bottom of my heart who were always present to help me out and make this project a success. Last but not the least, I would like to express my heartiest thanks and gratefulness to almighty God for his divine blessings, which helped me complete the final year project successfully This project was made possible because of inestimable inputs from everyone involved, directly or indirectly.

ABSTRACT

Communicating with the hearing-impaired person is always a great challenge. People with hearing disability uses hand gestures as a medium to communicate with others. By these hand gestures it is not necessary for the person to communicate vocally. There are various sign languages developed and used according to their native language. In this study, a system has been proposed to recognize Indian Sign Language (ISL) from hand gestures and generates text for the same. It is then converted to speech for effective understanding. A CNN based model is developed to recognize the Indian Sign Language with the hand gestures through computer vision in real-time. The model is trained on a dataset of ISL hand gestures and can recognize a wide range of gestures with high accuracy. The speech generation module converts the recognized gestures into speech using text-to-speech technology. The proposed system has potential applications in assisting the hearing-impaired community in India by providing them with an efficient and easy-to-use tool for communication. In addition, this system provides the paradigm of generating sign language from text, which enables communication in two ways without the use of an interpreter.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Networks
SRS	Software Requirement Specification
ISLR	Indian Sign Language Recognition
ASL	American Sign Language
ISL	Indian Sign Language

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Sign language is a visual communication used primarily by people with hearing disabilities. Instead of using acoustic sound patterns, sign language relies on gestures such as hand gestures, hand orientation, and facial emotions. Sign language can be used to communicate in situations when verbal communication is not possible, such as between speakers of mutually incomprehensible languages or when one or more communicators would be hearing impaired. This form of language is not universal and has various patterns depending on the people. People utilize a variety of sign languages all, Australian, French, Spanish, Chinese, Japanese, Indian, and many more are among the most popular sign languages. It is becoming more and more difficult for hearing-impaired people to communicate without an interpreter because many people in this world are not well versed with sign language, which makes them feel isolated. This recognition model for Sign Language has gained widespread acceptance as a means of communicating medium between people with hearing disabilities and ordinary people. Indian Sign Language (ISL) is a natural language used by the hearing-impaired community in India. It is a visual language that relies on hand gestures and facial expressions for communication. With the advancement of technology, many people show great interest in developing tools and systems that can aid in ISL recognition and communication.

Real-time methods for recognizing sign language are sensor-based and vision-based. The user of the sensor-based technique must put on gloves. Flex sensors, accelerometers, and motion sensors are built into the glove to track hand movement. A webcam is used in the vision-based system to collect photos of sign language, which are subsequently processed to identify the sign language. Many studies have been conducted in American Sign Language (ASL), while very few have been carried out in Indian Sign Language (ISL). With the help of Indian sign language (ISL), this proposed model tries to bridge the communication gap between hearing persons and the hearing-impaired persons. This model employs a CNN to recognize hand gestures using computer

vision to generate the corresponding characters, words, and sentences using finger spelling (used to spell words letter by letter). Latterly, CNN has shown great promise in image recognition tasks. By leveraging their ability to learn complex patterns and features from images, CNNs have been successfully applied to various recognition tasks. In this project, we propose an ISL recognition system based on CNN that can recognize a wide range of hand gestures with high accuracy. First, the sign images for ISL are captured using a webcam. The dataset consists of sign images for 26 alphabets, 0-9 digit and blank images for blank space as shown in figure 1. Second, the collected raw images are pre-processed and the feature is extracted. Third, the pre-processed image dataset of hand gestures is then trained using the proposed CNN based model to classify the images. Finally, the Indian sign language is recognized, and corresponding text is generated. To enable communication between hearing-impaired and blind persons, the created text is subsequently converted into speech to remove the communication barrier.

1.2 OBJECTIVE

The main objective of this project is to help the hearing-impaired people to communicate with normal people by using this automatic Indian Sign Language Recognition System with speech generation. The objective is to propose a model to recognize hand gestures in real time and generate not only a single character but also a whole sentence based on finger spelling method. After the text generation, it is converted to speech for better understanding and also to enhance communication between hearing-Impaired and visually impaired people. To enforce the two-way communication the hand gestures are also generated for the text or sentence, so that it will be useful for hearing impaired persons and the learners of sign language.

1.3 BACKGROUND OF THE STUDY

Sign language is a manual communication used primarily by people with hearing disabilities. Since most people are not familiar with sign language, it is getting harder and harder for hearing-impaired people to communicate without a translator, which makes them feel isolated. To

overcome this issue, we proposed a system with the help of Indian sign language (ISL), this proposed model tries to fill a communication gap between hearing- and hearing-impaired persons. This Indian Sign Language Recognition (ISLR) system helps to translate the sign language using hand gesture in real time and translate them into corresponding text and generate speech to bridge the communication gap between hearing impaired and the blind persons.

1.4 CHALLENGES

The Indian Sign Language Recognition System (ISLR) is a technology that aims to recognize and translate sign language into written or spoken language. However, developing an effective ISLR poses several challenges. Here are some of them: Variability in sign language: Sign language is not universal and varies from region to region. In this world, there are many different sign languages, such as Indian Sign Language, Chinese Sign Language, American Sign Language etc. This variability in sign language makes it challenging to develop an ISLR that can accurately recognize and translate the Indian sign languages.

Lack of standardization: Unlike spoken languages, sign languages are not standardized, and there are no clear rules for grammar and syntax. This makes it difficult to develop a robust and accurate recognition system.

Complex gestures: Sign language includes a variety of complex gestures, facial expressions, and body movements, which can be difficult to recognize and interpret accurately. For example, the meaning of a sign can change based on the direction of the sign or the position of the hand.

Limited data: Developing a machine learning based ISLR requires a large amount of data for training and testing. However, there is limited data available for Indian Sign Language, which makes it challenging to develop an accurate recognition system.

Limited hardware resources: Many areas in India do not have access to high-end hardware resources, such as GPUs, which are necessary for training and testing deep learning models. This can limit the development of ISLR in these regions.

Overall, developing an effective ISLR system that can accurately recognize and translate Indian Sign Language poses several challenges, including variability in sign language, lack of

standardization, complex gestures, limited data, and limited hardware resources.

1.5 APPLICATIONS

The real-time Indian sign language recognition system has a wide range of applications and uses, some of which are: The system can help people with hearing impairments to communicate with others in real-time using Indian sign language. The system can be used in schools and universities to help deaf students learn and communicate effectively with their peers and teachers. The system can improve accessibility for people with hearing impairments, enabling them to access public services, transportation, and other facilities. The system can be used as an assistive technology for people with disabilities, helping them to interact with technology more effectively. Overall, the real-time Indian sign language recognition system has the potential to improve the quality of life for people with hearing impairments, promote inclusivity and accessibility, and advance technological innovations.

CHAPTER 2

LITERATURE SURVEY

2.1 RESEARCH WORKS

1. Deep Learning Approach for Sign Language Recognition.

Bambang Krismono Triwijoyo [1] proposes a deep learning-based approach for recognizing sign language. The authors introduce a new dataset consisting of sign language gestures recorded by multiple users, and they use this dataset to train a deep convolutional neural network (CNN) model for recognizing sign language gestures. The proposed model achieved a high accuracy of 97.8% on the test dataset. The authors also compared their approach with several existing methods and showed that their proposed approach outperformed them in terms of accuracy. They concluded that deep learning-based approaches have great potential for sign language recognition and can be used in various applications, such as assistive technologies for the hard-of-hearing people. Overall, the paper presents a promising contribution to the field of sign language recognition and highlights the effectiveness of deep learning techniques in this domain.

2. Indian Sign Language recognition system using SURF with SVM and CNN.

Shagun Katoch [2] presents a sign language recognition system for Indian Sign Language (ISL) using the Speeded Up Robust Features (SURF) algorithm in combination with Support Vector Machines (SVM) and Convolutional Neural Networks (CNN). The authors introduced a new dataset for ISL gestures recorded by 30 participants and evaluated the proposed approach on this dataset. They extracted SURF features from the input images and used SVM and CNN for classification. The results showed that the proposed approach achieved an accuracy of 98.8% with SVM and 99.3% with CNN. The authors compared their approach with several existing methods and showed that their proposed approach outperformed them in terms of accuracy. They concluded that the proposed approach has great potential for ISL recognition and can be used in various applications, such as education and communication for the hearing-impaired. Overall, the paper

presents a promising contribution to the field of sign language recognition, specifically for Indian Sign Language, and highlights the effectiveness of combining feature extraction techniques with both SVM and CNN for classification.

3. Real time conversion of sign language to speech and prediction of gestures using Artificial Neural Network

Abey Abraham [3] proposes a real-time sign language recognition system that uses an Artificial Neural Network (ANN) for gesture prediction and conversion to speech. The system consists of two parts: gesture recognition using the ANN and speech synthesis using the eSpeak engine. The authors evaluated the system on a dataset of Indian Sign Language gestures and achieved an accuracy of 97.9% in recognizing the gestures. The proposed system has potential applications in improving communication for the hearing-impaired. The authors also discussed the limitations and future work to improve the accuracy of the system, such as incorporating other features and deep learning techniques.

4. Conversion of sign language into text.

Mahesh Kumar N B [4] proposes a system that converts American Sign Language (ASL) into text using a combination of image processing techniques and machine learning algorithms. The system uses OpenCV to process the input video of ASL gestures and extract features, and then applies a combination of K-means clustering and Artificial Neural Network (ANN) for classification. The proposed system achieved an accuracy of 95% in recognizing ASL gestures and converting them into text. The system has potential applications in improving communication for the deaf and hard-of-hearing and can be further improved by incorporating other features and techniques.

5. Conversion of Sign Language Video to Text and Speech

Mr. G. Sekhar Reddy [23] proposes a system for converting sign language video to text and speech, with the aim of improving communication between hard of hearing individuals and non-

signing individuals. The system uses computer vision and machine learning techniques for sign language recognition and natural language processing techniques for text and speech generation. The authors collected a dataset of sign language videos and their corresponding text transcriptions and evaluated the proposed system on this dataset. The results showed that the system achieved a high recognition accuracy rate of 94.4% and a high text-to-speech accuracy rate of 96.5%. The paper demonstrates the feasibility and effectiveness of using a combination of computer vision, machine learning, and natural language processing techniques for sign language communication. The proposed system has the potential to improve the accessibility and inclusivity of communication for hearing impaired individuals, particularly in settings where sign language interpretation is not readily available.

6. 3D – CNN based dynamic gesture recognition for Indian sign language Modeling

Dushyant Kumar Singh[10] proposes a system for recognizing dynamic hand gestures in the Indian Sign Language (ISL) using a 3D Convolutional Neural Network (3D-CNN) model. The proposed system captures and processes dynamic hand gestures in a video format, which is then classified using the 3D-CNN model and they collected a dataset of dynamic hand gestures in the ISL, which were recorded using a Kinect sensor. The dataset was preprocessed, and 3D volumes of each gesture were generated. These volumes were then used to train a 3D-CNN model, which was able to achieve an accuracy of 97.6% in recognizing dynamic hand gestures. They also compared the performance of their 3D-CNN model with other machine learning models such as SVM and k-NN. The results showed that the 3D-CNN model outperformed these models in terms of accuracy.

7. Sign Language Recognition System Using TensorFlow Object Detection API

Sharvani Srivastava[12] presents a system for recognizing Indian Sign Language (ISL) gestures using the TensorFlow Object Detection API. The proposed system uses a camera to capture hand gestures and the TensorFlow Object Detection API to recognize the corresponding

sign language word or phrase and they collected a dataset of ISL gestures and preprocessed it to extract features. They then used these features to train a custom object detection model using the TensorFlow Object Detection API. The model was able to achieve an accuracy of 97.34% in recognizing ISL gestures. They also integrated a real-time video feed to capture hand gestures and process them using the trained model. The system was able to recognize the ISL gestures in real-time and display the corresponding text on the screen.

8. Indian Sign Language Communicator Using Convolutional Neural Network

Arvind Sreenivas[14] presents a system for recognizing and translating Indian Sign Language (ISL) gestures into text and speech using Convolutional Neural Network (CNN) models. The proposed system consists of a camera to capture hand gestures, a CNN model for recognition and a text-to-speech conversion system for generating spoken output and they collected a dataset of ISL gestures and preprocessed it to extract features, then used these features to train a CNN model, which includes multiple convolutional layers and a pooling layer. The CNN model was able to achieve an accuracy of 97.78% in recognizing ISL gestures. They also integrated a text-to-speech conversion system to generate spoken output based on the recognized gestures. The system was able to translate the recognized ISL gestures into English text and then convert it to speech.

9. Design of Sign Language Recognition Using E-CNN

Citra Suardi[24] presents a system for recognizing sign language gestures using an Enhanced Convolutional Neural Network (E-CNN) and propose a system that uses a camera to capture hand gestures and an E-CNN model to recognize the corresponding sign language word or phrase. The system is designed to recognize signs from the Indonesian sign language alphabet.

They collected a dataset of sign language gestures and preprocessed it to extract features. They then used these features to train the E-CNN model, which includes multiple convolutional layers and a pooling layer. The E-CNN model was able to achieve an accuracy of 96.7% in recognizing sign language gestures and also compared the performance of their E-CNN model with other machine learning models such as SVM, k-NN, and Random Forest. The results showed that the E-CNN model outperformed these models in terms of accuracy. The high accuracy of the system also

suggests that it could be used in real-world applications such as assistive technology for the hearing-impaired people.

10. Sign language recognition using machine learning algorithm

Radha S. Shirbhate[28] presents a system for recognizing sign language gestures using machine learning techniques and propose a system that uses a camera to capture hand gestures and machine learning algorithms to recognize the corresponding sign language word or phrase. The system is designed to recognize signs from the Indian sign language alphabet and collected a dataset of sign language gestures and preprocessed it to extract features. They then used these features to train machine learning models such as Support Vector Machine (SVM), Random Forest (RF), and k-Nearest Neighbors (k-NN) classifiers. The results showed that the SVM model performed the best in recognizing sign language gestures with an accuracy of 92.5%.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 HARDWARE SPECIFICATIONS

The hardware requirements for an Indian Sign Language Recognition (ISLR) System depend on the specific approach and technology used for the recognition task. Here are some general hardware requirements to consider:

Processor: CPU (minimum i3 processor) or GPU (for speeding u the process).

Memory: At least 4 GB of RAM is recommended (8GB can be used for better performance).

Camera: A high-quality camera is required to capture the sign language gestures accurately. A camera with at least 720p resolution is recommended.

Storage: A large amount of storage space is required to store the datasets and trained models. At least 100 GB of storage space is recommended.

3.2 SOFTWARE SPECIFICATIONS

To develop the ISLR system using machine learning, the following software and packages are required.

Python: Version 3.10.10 or higher

Visual Studio: Version 1.77.3 or higher

Packages Used: TensorFlow, Keras, Tkinter (for GUI window), OpenCV (for image capturing), Pyttsx3 (for Speech generation)

3.3 SOFTWARE REQUIREMENT SPECIFICATION

An SRS is a description of a software system to be developed. The software requirements specification document lists sufficient and necessary requirements for the software development.

Python: Python is consistent and is anchored on simplicity, which makes it most appropriate for machine learning. The Python programming language best fits machine learning due to its independent platform and its popularity in the programming community.

TensorFlow: TensorFlow is an end-to-end, open-source machine learning platform. You can think of it as an infrastructure layer for differentiable programming. It combines four key abilities: Efficiently executing low-level tensor operations on CPU, GPU, or TPU. Computing the gradient of arbitrary differentiable expressions. Scaling computation to many devices, such as clusters of hundreds of GPUs. Exporting programs ("graphs") to external runtimes such as servers, browsers, mobile and embedded devices.

Keras: Keras is the high-level API of the TensorFlow platform: an approachable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning neural networks. It provides essential abstractions and building blocks for developing and shipping machine learning solutions with high iteration velocity.

Tkinter: Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Pyttsx3: pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python 2 and 3.

CHAPTER 4

SYSTEM ANALYSIS

4.1 EXISTING SYSTEM

- There exists numerous sign language recognition system for American Sign Language (ASL), Chinese Sign Language (CSL), Australian Sign Language (Auslan), French Sign Language (LSF), etc., but only very few systems available for Indian Sign Language (ISL).These systems use various techniques such as computer vision, machine learning, and sensor-based technologies to recognize and interpret hand gestures.
- Most of the existing sign language recognition systems can recognize only few alphabetical characters or digits. Only a few of them can recognize all the 26 alphabets of the English language. This limitation is due to the complexity involved in recognizing and interpreting the various hand gestures and movements involved in forming words and sentences in sign language.
- The existing sign language recognition systems do not have the feature of speech output for Indian Sign Language, which means they can only recognize and interpret sign language gestures and convert them to text.
- In the case of Sign Language Recognition (SLR) systems, there exists only a one-way communication system, which means the existing systems can only recognize and interpret sign language gestures and convert them to text. There is no system available yet that can convert text into sign language gestures.
- There are two main types of sign language recognition systems: vision-based and sensor-based. Vision-based systems use cameras and computer vision techniques to recognize and interpret sign language gestures. Sensor-based systems, on the other hand, use sensors attached to the user's hand and wrist to capture the movements of the hand and interpret the sign language gestures. Both these models can be used for hand gesture-based sign language recognition system. However, the choice of the model depends on the application, accuracy requirements, and cost.

4.2 PROPOSED SYSTEM

- The proposed system is designed specifically for Indian Sign Language, which is used by the hearing-impaired community in India. Unlike American Sign Language, Indian Sign Language primarily uses double-handed gestures to convey meaning. Therefore, the proposed system is designed to recognize and interpret these gestures accurately.
- Unlike most existing sign language recognition systems, which can only recognize single letters or digits, the proposed system is designed to recognize and interpret complete words and sentences in Indian Sign Language. This makes it easier for the hearing-impaired people to communicate effectively with others.
- The proposed system not only recognizes and interprets Indian Sign Language gestures but also includes speech generation. This means that the system can convert text or sign language gestures into spoken language, making communication easier and more effective between the hearing community and visually impaired community.
- The proposed system is designed to allow two-way communication between the hearing and hard-of-hearing communities. This means that the system can recognize and interpret sign language gestures and convert them to text, as well as convert text into sign language gestures along with voice output for each shown gesture.
- The proposed system is designed to recognize and interpret all 26 alphabetical characters and digits in Indian Sign Language. This makes it more comprehensive and useful than most existing sign language recognition systems, which can only recognize a limited number of gestures.
- Overall, the proposed system is designed to provide a comprehensive and effective solution for sign language recognition and communication in Indian Sign Language, with the ability to recognize and interpret double-handed gestures, generate speech, and enable two-way communication between the hearing- and hearing-impaired communities.

4.3 NOVELTY IN PROPOSED WORK

- It can recognize all the alphabetical characters and digits for ISL.
- It also includes generation of words and sentences.
- It also includes speech output for generated texts.
- This proposed system includes two-way communication for Indian sign language system.

4.4 ALGORITHM

4.4.1 CONVOLUTIONAL NEURAL NETWORK (CNN)

The algorithm used for classifying the hand gesture is Convolutional Neural network (CNN). Convolutional Neural Network (CNN) is a type of neural network that is widely used in image recognition and classification tasks. It is inspired by the structure and function of the visual cortex in the human brain, which processes visual information in a hierarchical manner. CNNs consist of multiple layers, each performing a specific function in the image recognition process. The first layer is usually a convolutional layer, which extracts features such as edges, corners, and shapes from the input image. The second layer is typically a pooling layer, which reduces the spatial size of the feature maps while retaining the most important features. The third layer is usually a fully connected layer, which learns to classify the input image into different classes.

The benefits of CNNs include their ability to learn features directly from raw input data, their scalability to handle large datasets and complex tasks, and their high accuracy in image recognition and classification tasks. CNNs are also able to handle variations in the input data, such as changes in lighting, scale, and orientation, which makes them well-suited for real-world applications.

CNNs are used in a wide range of applications, including object recognition, facial recognition, medical image analysis, and autonomous driving. In object recognition, CNNs can be used to identify and classify different objects in an image, such as cars, animals, and buildings. In facial recognition, CNNs can be used to identify individuals based on their facial features, which has applications in security and law enforcement. In medical image analysis, CNNs can be used to

detect and diagnose different medical conditions, such as tumors and abnormalities. In autonomous driving, CNNs can be used to recognize and classify different objects in the environment, such as other vehicles, pedestrians, and traffic signs.

Overall, CNNs are a powerful tool for image recognition and classification tasks, with a wide range of potential applications in various fields. Their ability to learn directly from raw input data and handle variations in the input make them well-suited for real-world applications where input data can be complex and diverse.

CHAPTER 5

DESIGN AND IMPLEMENTATION

5.1 METHODOLOGIES

The Overview of the proposed ISLR system's process flow is as follows. The data collection process is the first stage of the proposed system. The proposed system's dataset consists of hand gesture images for alphabets and numbers. The following stage is data pre-processing. It involves converting the raw dataset to a grayscale image and applying a Gaussian blur filter to that image to extract its features during the data pre-processing stage. The next stage is the training phase. The proposed CNN-based model is trained to categorize images using hand gesture features extracted during the pre-processing stage. The proposed model includes three convolutional layers to extract high-level features from hand gestures, three pooling layers with ReLU activation functions to reduce dimensionality while preserving the most important feature, and a fully connected layer to classify the hand gestures. The following stage is gesture recognition. In this stage, the system uses OpenCV to capture the live camera feed, then pre-processes the shown gesture, and finally predicts the hand gesture using the trained proposed model. The next stage is sentence formation based on the recognized individual hand gesture. Next, the speech is generated for the sentence formed. The proposed ISLR system includes an additional module for converting text to sign that converts the text into equivalent hand gesture.

5.1.1 DATA COLLECTION

The first step is the data collection for Indian Sign Language Recognition System (ISLR). Because of a lack of study in this area, an appropriate data set for ISL is not yet readily available. To fill in the gaps left by unavailable data; we have generated the own dataset using data from a variety of sources. The dataset consists of images for alphabets (A-Z), numbers (0-9) and blank images for space in a sentence.

Therefore, the total collection has 37 categories. Each category comprises of 1200 hand gesture images along with the labels corresponding to them. Figure 1 shows sample dataset for the proposed ISL system.

Figure 1. Sample Dataset

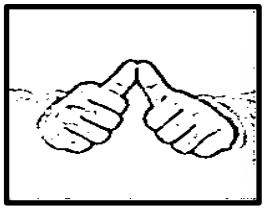
5.1.2 DATA PRE-PROCESSING

The input image must first be converted to grayscale. Grayscale conversion simplifies the image by reducing the color information to a single channel, making it easier to extract the important features of the hand gesture. Next, the grayscale image is then blurred using a Gaussian filter to lessen noise and smoothen the edges of the hand region as seen in figure 2. This step is required to improve the efficiency of the hand recognition algorithm and reduce false positives.

Finally, the hand region is then separated from the background using thresholding. Thresholding converts the grayscale hand gesture image into a binary image, where pixels with values above a certain threshold are set to white, and those below the threshold are set to black. The threshold value can be determined using various techniques such as Otsu's method, which selects the threshold value that minimizes the variance within the hand region.

Raw Image

Gray Scale Image



After applying
Gaussian filter

Figure 2. Hand Gesture Pre-processing

5.1.3 CLASSIFICATION

Once the ROI has been identified, we then use the deep learning techniques for training the image dataset for classification. Classification of hand gestures is an important step in the Indian Sign Language recognition system. It involves identifying a specific gesture made by the user and matching that gesture with the corresponding word or phrase in sign language. CNNs are a popular technique for gesture recognition since they can learn complex features from the input images of hand gestures and achieve high accuracy.

PROPOSED CONVOLUTIONAL NEURAL NETWORKS (CNN)

The proposed system consists of Three convolutional layers with ReLU activation functions, followed by max pooling layers, a flatten layer, and a fully connected layer with a SoftMax activation function make up the proposed CNN architecture in figure 3 for ISLR utilizing hand gestures.

High-level features from the input image are extracted by the three convolutional layers. A ReLU activation function that provides non-linearity into the network and enables the network to

learn more complicated features comes after each convolutional layer. A max pooling layer is applied after each convolutional layer to decrease the feature maps' spatial dimensions while retaining the most crucial characteristics.

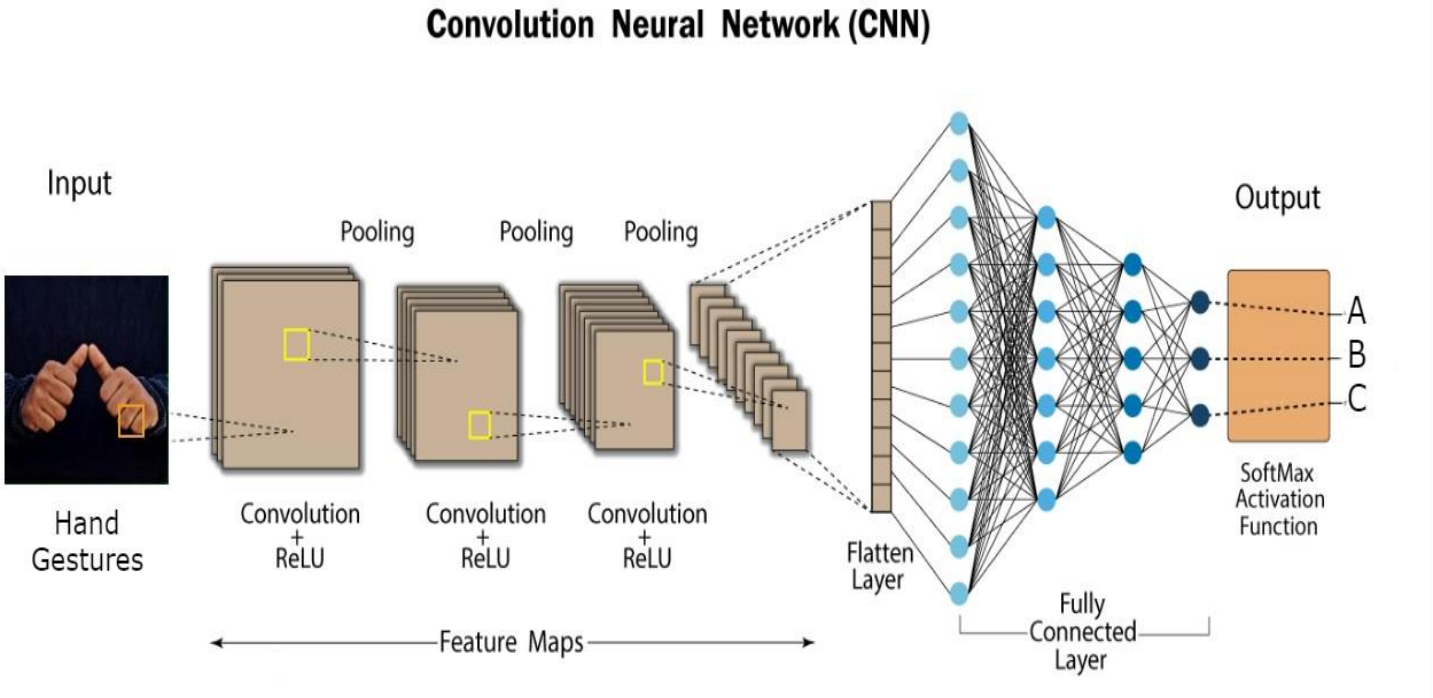


Figure 3. Proposed CNN Architecture

To feed the fully connected layer for classification, the output of the last max pooling layer must first be flattened into a one-dimensional vector. The fully connected layer features a SoftMax activation function that provides a probability distribution over the various classes and normalizes the layer's output.

Table 1.Proposed CNN layer Architecture

Layer	Type	Output Shape
Convolutional Layer 1	Con2d+ReLU	(None, 126, 126, 64)
	Max_pooling2d	(None, 63, 63, 64)
Convolutional Layer 2	Con2d+ReLU	(None, 61, 61, 64)
	Max_pooling2d_1	(None, 30, 30, 64)
Convolutional Layer 3	Con2d+ReLU	(None, 28, 28, 64)
	Max_pooling2d_2	(None, 14, 14, 64)
Flatten	Flatten	(None, 12544)
Dense	Dense	(None, 128)
Dropout	Dropout	(None, 128)
Dense	Dense_1	(None, 36)

Overall, this CNN architecture is well-suited for the proposed system of Indian sign language recognition using hand gestures. The multiple convolutional layers with ReLU activation functions allow the network to learn complex features, while the max pooling layers and flatten layer lessen the dimensionality of the feature maps and produce a one-dimensional vector for classification. Ultimately, the SoftMax activation function in the fully connected layer creates a probability distribution over the various classes, enabling the network to produce accurate predictions.

5.1.4 GESTURE RECOGNITION

In the proposed model, the hand gesture is recognized using ROI which is a popular technique for accurately identifying and classifying hand gestures. ROI refers to a specific region of the image where the hand gesture is expected to occur. By focusing on the ROI, the system can improve the accuracy of gesture recognition while reducing computational complexity.

The ROI is identified using the skin color detection technique. Skin color detection is a common technique used to identify the ROI since the hand is typically a different color than the background. This technique involves applying a color filter to the input image and extracting the pixels that fall within a certain color range. To separate the hand region from the background, threshold is applied to the generated image.

5.1.5 SENTENCE FORMATION

In the proposed system, the corresponding text is generated by recognizing the hand gesture and sentence is formed using finger spelling method. Finger spelling recognition is an important component of Indian Sign Language (ISL) recognition, as it enables the system to recognize words and phrases that do not have specific hand gestures. Fingerspelling is the process of spelling out words letter-by-letter using hand gestures, and it is a common method of communication for the hearing-impaired community.

In the proposed system, a letter is added to the current text if its count reaches a certain threshold number, and no other letters are comparable to it. The system displays alternative characters to add to the existing text if comparable letters are found. If the system notices a blank screen without a hand signal, it prints the space. The phrase is finally constructed by letter-by-letter recognition of the alphabetic hand gesture as seen in figure 6.

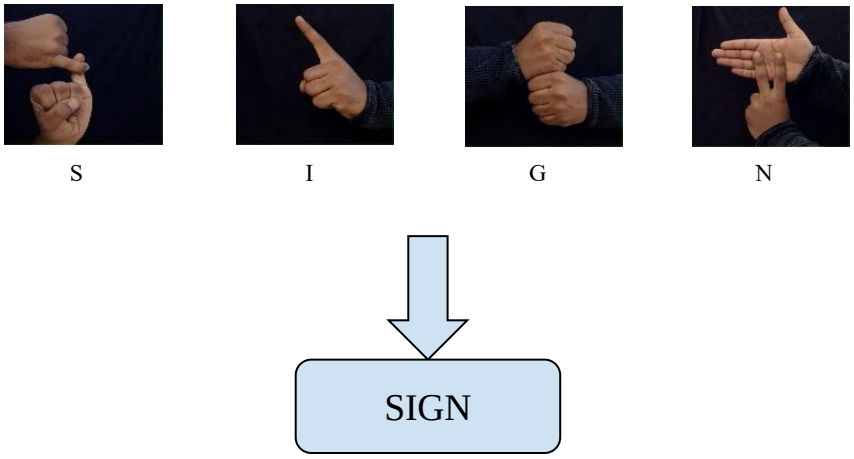


Figure 4. Sentence Formation using Finger Spelling Method

5.1.6 SPEECH GENERATION

In the proposed system, the text-to-speech conversion process enables the system to provide spoken language output for the sentence formed by detecting sign language gestures.

Text to speech systems work by analyzing the text input and breaking it down into individual phonemes, or units of sound. The system then uses a speech synthesis engine to generate the corresponding speech waveform for each phoneme, which is then combined to produce the final spoken output. The text to speech system uses python text to speech library (pyttsx3) to generate speech from text. Using the pyttsx library, developers can generate speech output in a variety of languages and accents, and customize the speed, volume, and pitch of the generated speech. The library also supports call-backs for events such as the start and end of speech, making it easy to integrate with other Python libraries and applications. This can be especially useful for communication between the people with hearing impairment and the blind, as it enables them to communicate more effectively. Overall, text-to-speech conversion is an essential component of SLR systems, as it enables the system to offer hearing-impaired people a systematic means of communication.

5.1.7 TEXT TO SIGN CONVERSION

The system enables two-way communication by providing the additional feature of converting text into the corresponding hand gesture as shown in figure 5. Text-to-sign conversion is a crucial component of Indian Sign Language (ISL) recognition systems as it enables the system to convert text into sign language gestures. The process involves converting the text input into sign language gestures by searching and retrieving the hand gesture from the classified dataset. Additionally, the system includes the voice features for the displayed hand gesture.

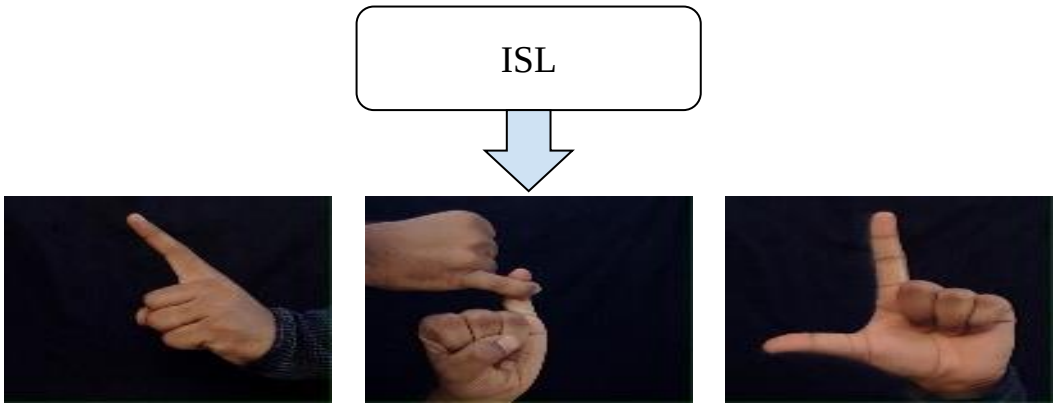


Figure 5. Text to Sign Language Conversion

5.1.8 EXPERIMENTS AND RESULTS

Images of hand movements corresponding to the alphabets and numeric symbols used in Indian Sign Language make up the dataset utilized in the proposed ISLR System. The collected data is intended to aid in the development of a proposed CNN-based model that is capable of effectively identifying and interpreting these gestures.

The dataset includes a total of 37 classes, representing the 26 alphabets, 10 digits and a blank image for blank space. The dataset is captured using a high-resolution camera that records the hand gestures at 30 frames per second, providing a rich and diverse set of datasets of hand gestures for training and testing the proposed model. The images and videos are captured in RGB format,

providing colour information that can be used to improve the efficiency of the recognition model.

In data pre-processing, the process of Grayscale conversion removes the colour information from the images, resulting in a higher contrast and clearer edges, which can help the model to better distinguish between the different hand gestures. Gaussian blur can help to smooth out the images and reduce noise, making it easier for the model to identify the important features in the images.

By reducing the noise and improving the quality of the images, pre-processing techniques like grayscale conversion and Gaussian blur can make it easier and faster for the proposed model to train on the dataset and to make accurate predictions.

Pre-processing the dataset in this way helps to reduce overfitting by removing unnecessary details from the images and focusing on the most important features, which leads to better generalization and more accurate predictions on new, unseen data.

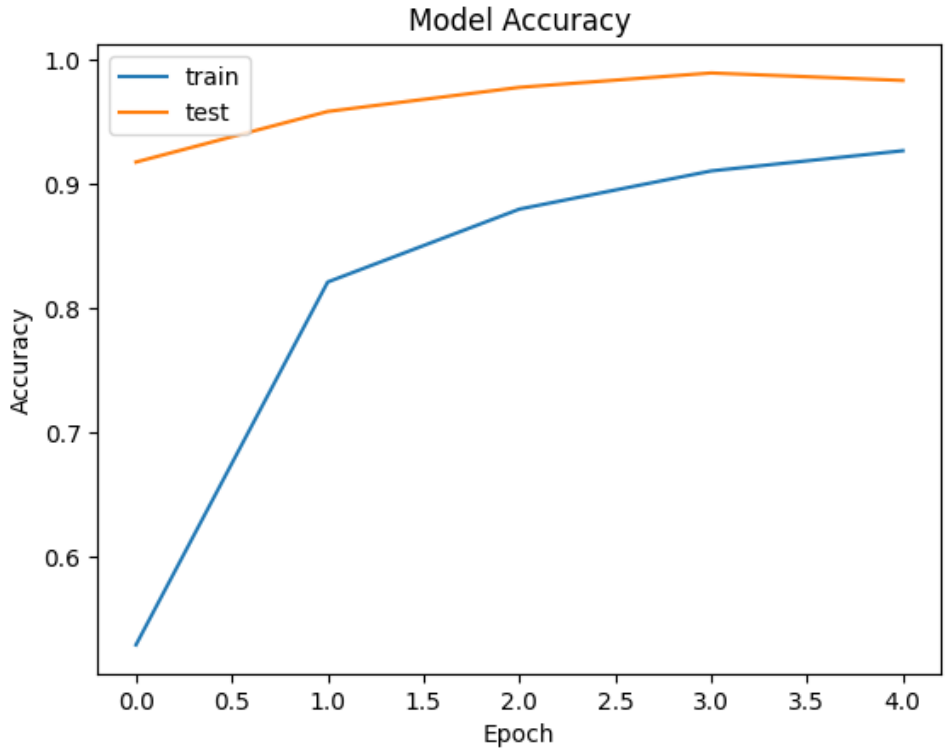


Figure 6. Accuracy graph for proposed CNN

The use of hand gestures in Sign Language Recognition employs a variety of algorithms to train the dataset. Here are some comparisons of popular algorithms.

With an accuracy of 85.45%, Sharvani Srivastava[12] proposed a TensorFlow Object Detection API method for Sign Language Recognition System . Jie Huang[13] proposed a Video-Based Sign Language Recognition without Temporal Segmentation using the method of LS-HAN and 3D CNN with an accuracy rate of 82.7%. Mandeep Kaur Ahuja[15] suggested a Principal Component Analysis method for Hand Gesture Recognition with an accuracy of 91.43%. Neel Kamal Bhagat[25] propose a Indian Sign Language Gesture Recognition using Image Processing and Deep Learning and a method of Convolutional Neural Network , LSTMs , Microsoft Kinect with an accuracy of 78.3%. Tulay Karayilan[22] proposed a Sign Language Recognition using a method ANN(Backpropagation Algorithm) with accuracy rate of 85.7%. Fayed F.M. Ghaleb[11] using a CRF and SVM method for Vision-Based Hand Gesture Spotting and Recognition with an accuracy rate of 92.50%.

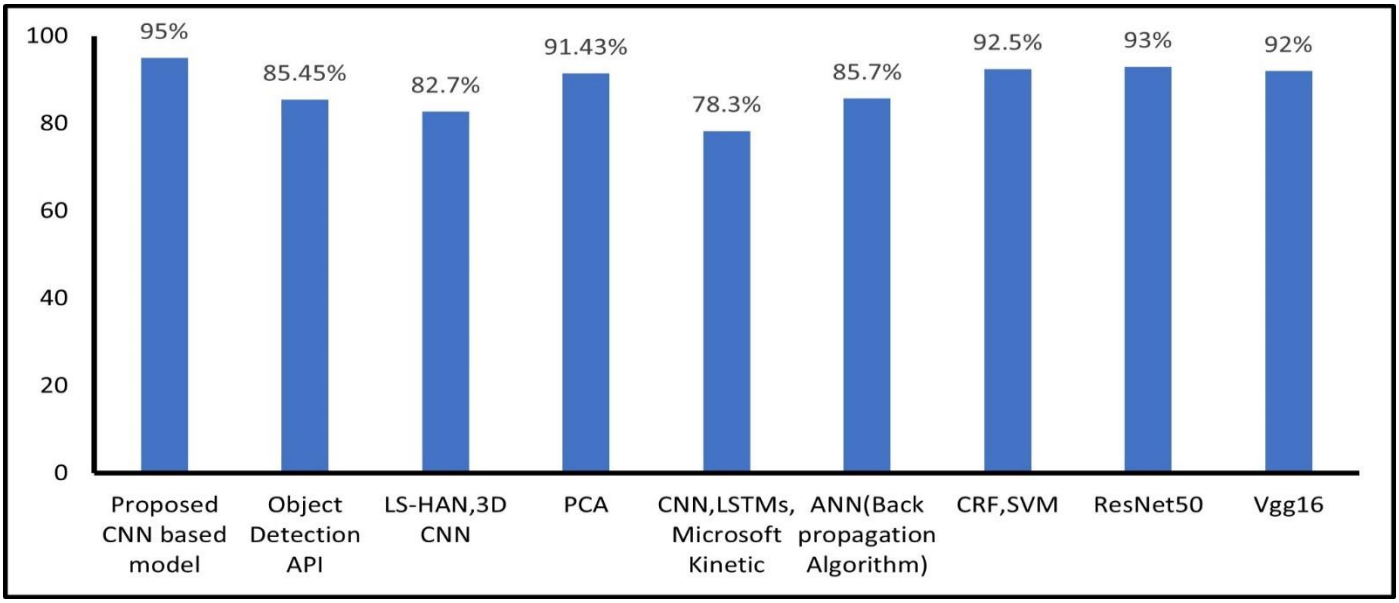


Figure 7. Model Accuracy (in %) Comparisons for various algorithm

This proposed technique makes use of a CNN architecture with three convolutional layers with 32,

64 filters, respectively. Each convolutional layer is followed by max pooling layer that have ReLU activation functions. The output of the third max pooling layer is flattened and fed into a dense layer with 128 units, and then fully connected layer that has a SoftMax activation function. The final output layer has 37 units, one for each class. The main benefit of using categorical Cross-Entropy is because it is designed to handle multi-class classification tasks, where each sample can belong to one of several classes. Here Adam optimizer is used in this training process.

Adam (Adaptive Moment Estimation) is an optimization algorithm that updates model parameters by taking a moving average of the gradient and the squared gradient. It also employs bias correction to correct for the bias caused by the moving averages' initial estimates. Adam is used for the proposed model because it is an efficient optimization algorithm for training the deep learning model and it can handle sparse gradients.

After training the proposed model for 5 epochs, the proposed model achieved an accuracy of 95% on the validation set with minimum loss.

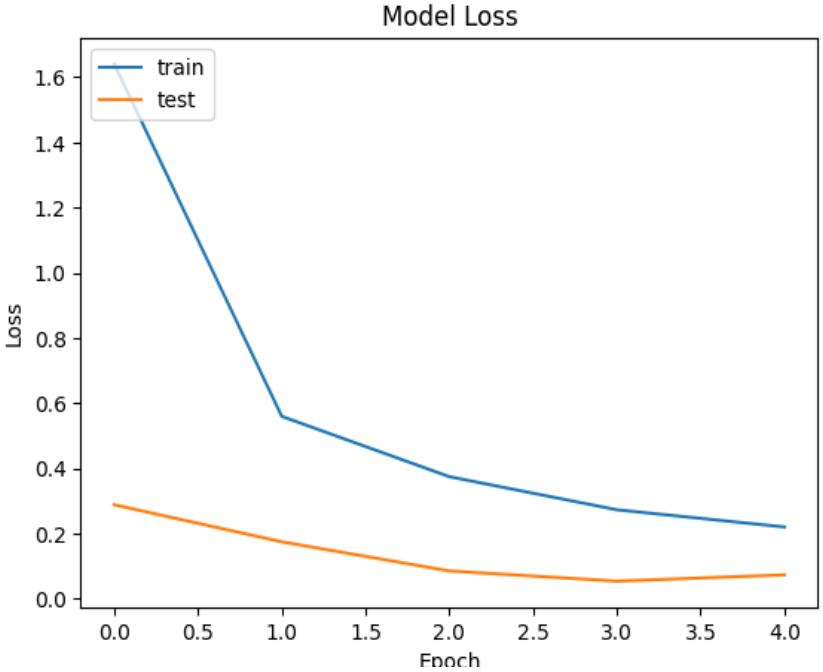


Figure 8. Loss graph for proposed CNN

5.2 SYSTEM ARCHITECTURE DIAGRAM

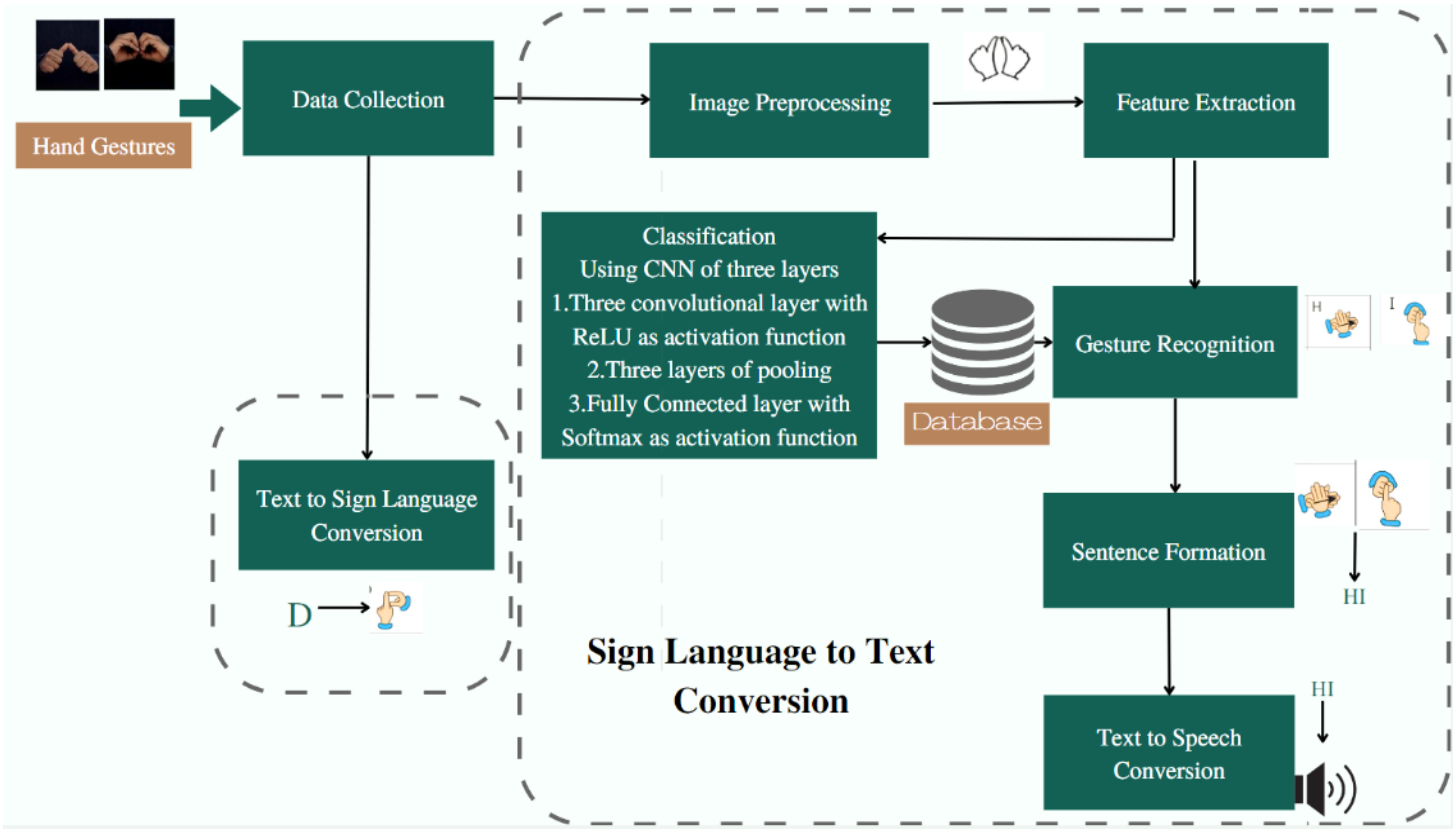


Figure 6. ISLR System Architecture

CHAPTER 6

CONCLUSION AND FUTURE WORK

This system for Indian Sign Language Recognition proves the effectiveness of the proposed model for recognizing ISL gestures and generating spoken output from the recognized gestures and the feature of text to sign language conversion. The proposed Indian Sign Language Recognition System using hand gestures achieves the accuracy of 95%. However, there is still a need for further research to improve the accuracy and robustness of these systems, as well as to develop more advanced speech generation systems for sign language recognition. While current ISL recognition systems focus primarily on hand gesture recognition and conversion of text to sign language, there is potential to integrate other modalities, such as real time handwritten text to sign language conversion to improve the ISLR System. Future research could explore the use of multi-modal recognition systems that combine hand gesture recognition with other modalities.

APPENDICES

SAMPLE CODE

Image_preprocessing.py:

```
# CONVERTING RGB IMAGES TO BLACK AND WHITE IMAGES

import numpy as np
import cv2
minValue = 70
def func(path):
    frame = cv2.imread(path)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    blur = cv2.GaussianBlur(gray, (5, 5), 2)

    _, res = cv2.threshold(blur, minValue, 255, cv2.THRESH_BINARY_INV +
cv2.THRESH_OTSU)
    return res
```

preprocessing.py:

```
# STEP 2 (PART2): IMAGE PREPROCESSING
import numpy as np
import cv2
import os
from image_processing import func
if not os.path.exists("myProcessData"):
    os.makedirs("myProcessData")
if not os.path.exists("myProcessData/train"):
    os.makedirs("myProcessData/train")
if not os.path.exists("myProcessData/test"):
    os.makedirs("myProcessData/test")
path = "myData/train"
output = "myProcessData"
a = ["label"]
for i in range(128 * 128):
    a.append("pixel" + str(i))
# outputLine = a.tolist()
```

```
label = 0
var = 0
c1 = 0
c2 = 0
for (dirpath, dirnames, filenames) in os.walk(path):
    for dirname in dirnames:
        print(dirname)
        for (direcpath, direcnames, files) in os.walk(path + "/" + dirname):
            if not os.path.exists(output + "/train/" + dirname):
                os.makedirs(output + "/train/" + dirname)
            if not os.path.exists(output + "/test/" + dirname):
                os.makedirs(output + "/test/" + dirname)
            num = 0.70 * len(files)
            i = 0
            for file in files:
                var += 1
                actual_path = path + "/" + dirname + "/" + file
                output_path = output + "/" + "train/" + dirname + "/" + file
                output_path_test = output + "/" + "test/" + dirname + "/" + file
                img = cv2.imread(actual_path, 0)
                bw_image = func(actual_path)
                if i < num:
                    c1 += 1
                    cv2.imwrite(output_path, bw_image)
                else:
                    c2 += 1
                    cv2.imwrite(output_path_test, bw_image)
                i = i + 1
        label = label + 1
print(var)
print(c1)
print(c2)
```

train.py:

```
# Importing the Keras libraries and packages
from keras.preprocessing.image import ImageDataGenerator
```

```
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense, Dropout
import os
#os.environ["CUDA_VISIBLE_DEVICES"] = "1"
sz = 128
# Step 1 - Building the CNN
# Initializing the CNN
classifier = Sequential()

# First convolution layer and pooling
classifier.add(Convolution2D(64, (3, 3), input_shape=(sz, sz, 1), activation="relu"))
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Second convolution layer and pooling
classifier.add(Convolution2D(64, (3, 3), activation="relu"))
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# third convolution layer and pooling
classifier.add(Convolution2D(64, (3, 3), activation="relu"))
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Flattening the layers
classifier.add(Flatten())

# Adding a fully connected layer
classifier.add(Dense(units=128, activation="relu"))
classifier.add(Dropout(0.5))

#classifier.add(Dense(units=36, activation="relu"))
#classifier.add(Dropout(0.5))

# softmax for more than 2 outputs neuron
classifier.add(Dense(units=36, activation="softmax"))
```

```
# Compiling the CNN
classifier.compile(
    optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"]
)  # categorical_crossentropy for more than 2


# Step 2 - Preparing the train/test data and training the model
classifier.summary()

train_datagen = ImageDataGenerator(
    rescale=1.0 / 255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True
)
test_datagen = ImageDataGenerator(rescale=1.0 / 255)

training_set = train_datagen.flow_from_directory(
    "myProcessdata/train",
    target_size=(sz, sz),
    batch_size=1,
    color_mode="grayscale",
    class_mode="categorical",
)
test_set = test_datagen.flow_from_directory(
    "myProcessData/test",
    target_size=(sz, sz),
    batch_size=1,
    color_mode="grayscale",
    class_mode="categorical",
)
classifier.fit(
    training_set,
    steps_per_epoch=5040,  # No of images in training set
    epochs=5,
    validation_data=test_set,
    validation_steps=2160,
)  # No of images in test set

# Saving the model
```

```
model_json = classifier.to_json()
with open("model_az.json", "w") as json_file:
    json_file.write(model_json)
print("Model Saved")
classifier.save_weights("model_az.h5")
print("Model saved")
```

app.py:

```
from tkinter.constants import COMMAND
from PIL import Image, ImageTk
import tkinter as tk
import cv2
from keras.models import model_from_json
import operator
from string import ascii_uppercase, digits
import pyttsx3
import string
import subprocess

class Application:
    def __init__(self):

        # Setup GUI
        self.root = tk.Tk()
        self.root.title("Indian Sign Language Recognition")
        self.root.config(background="#11754E")
        self.root.protocol("WM_DELETE_WINDOW", self.destructor)

        self.directory = "model/"

        self.vs = cv2.VideoCapture(cv2.CAP_DSHOW)
        # self.vs.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
        # self.vs.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)

        self.current_image = None
        self.current_image2 = None
```

```
self.json_file = open(self.directory + "model_az.json", "r")
self.model_json = self.json_file.read()
self.json_file.close()
self.loaded_model = model_from_json(self.model_json)
self.loaded_model.load_weights(self.directory + "model_az.h5")
self.ct = {}
self.ct["blank"] = 0
self.blank_flag = 0
for i in digits:#changes done
    self.ct[str(i)] = 0
for i in ascii_uppercase:
    self.ct[i] = 0
print("Loaded model from disk")

def talk():
    engine = pyttsx3.init()
    engine.say(self.str)
    engine.runAndWait()

def clearWord():
    self.word = ""

def clearSentence():
    self.str = ""

def text_Sign():
    #window = tk.TK()
    subprocess.call(["python", "texttosign.py"])
    #window.destroy()

self.root.geometry("900x1100")
self.panel = tk.Label(self.root)
self.panel.place(x=135, y=10, width=640, height=480)
self.panel2 = tk.Label(self.root)
self.panel2.place(x=950, y=20, width=310, height=310)
self.panel3 = tk.Label(self.root)
```

```
        self.panel3.place(x=350, y=560)
        self.T1 = tk.Label(self.root)
        self.T1.place(x=145, y=560)
        self.T1.config(text="Character :", font=("Comic Sans MS", 16, "bold"),bg="#11754E")

        self.panel4 = tk.Label(self.root)
        self.panel4.place(x=350, y=600)

        self.T2 = tk.Label(self.root)
        self.T2.place(x=145, y=600)
        self.T2.config(text="Word :", font=("Comic Sans MS", 16, "bold"),bg="#11754E")

        self.panel5 = tk.Label(self.root)
        self.panel5.place(x=350, y=640)

        self.T3 = tk.Label(self.root)
        self.T3.place(x=145, y=640)
        self.T3.config(text="Sentence :", font=("Comic Sans MS", 16, "bold"),bg="#11754E")

        my_button = tk.Button(self.root, text="speak", font=("Comic Sans MS", 10, "bold"),
bg="#E48F1B", fg="#34262B", command=talk)
        my_button.place(x=620, y=640)

        clear_button = tk.Button(self.root, text="Clear", font=("Comic Sans MS", 10, "bold"),
bg="#E48F1B", fg="#34262B", command=clearWord)
        clear_button.place(x=720, y=600)

        clear_button1 = tk.Button(self.root, text="Clear", font=("Comic Sans MS", 10, "bold"),
bg="#E48F1B", fg="#34262B", command=clearSentence)
        clear_button1.place(x=720, y=640)

        tts_button = tk.Button(self.root, text="Text to Sign Conversion", font=("Comic Sans MS",
14, "bold"), bg="#E48F1B", fg="#34262B", command=text_Sign)
        tts_button.place(x=1000, y=640)
        #tts_button.pack()
        self.str = ""
        self.word = ""
```

```
        self.current_symbol = "Empty"
        self.photo = "Empty"
        self.video_loop()

    def video_loop(self):
        ok, frame = self.vs.read()

        if ok:
            cv2image = cv2.flip(frame, 1)
            x1 = int(0.5 * frame.shape[1])
            y1 = 10
            x2 = frame.shape[1] - 10
            y2 = int(0.5 * frame.shape[1])
            cv2.rectangle(cv2image, (x1, y1 + 1), (x2, y2 - 1), (255, 0, 0), 1)
            cv2image = cv2.cvtColor(cv2image, cv2.COLOR_BGR2RGBA)
            self.current_image = Image.fromarray(cv2image)
            imgtk = ImageTk.PhotoImage(image=self.current_image)
            self.panel.imgtk = imgtk
            self.panel.config(image=imgtk)
            cv2image = cv2image[y1 : x1 + 1, y2 : x2 - 1]
            cv2image = cv2.resize(cv2image, (128, 128))
            gray = cv2.cvtColor(cv2image, cv2.COLOR_BGR2GRAY)
            blur = cv2.GaussianBlur(gray, (5, 5), 0)
            ret, res = cv2.threshold(
                blur, 90, 255, cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU
            )
            self.predict(res)
            self.current_image2 = Image.fromarray(res)
            imgtk = ImageTk.PhotoImage(image=self.current_image2)
            self.panel2.imgtk = imgtk
            self.panel2.config(image=imgtk)
            self.panel3.config(text=self.current_symbol, font=("Comic Sans MS", 10),
bg="#E0E0E0")
            self.panel4.config(text=self.word, font=("Comic Sans MS", 10),bg="#E0E0E0")
            self.panel5.config(text=self.str, font=("Comic Sans MS", 10),bg="#E0E0E0")
        self.root.after(17, self.video_loop)
```

```
    def predict(self, test_image):
        test_image = cv2.resize(test_image, (128, 128))
        result = self.loaded_model.predict(test_image.reshape(1, 128, 128, 1))
        prediction = {}
        prediction["blank"] = result[0][0]
        index = 0 #changes done

        for i in ''.join([str(digit) for digit in range(10)] + list(string.ascii_uppercase)): #changes done
            prediction[i] = result[0][index]
            index += 1

        # LAYER 1
        prediction = sorted(
            prediction.items(), key=operator.itemgetter(1), reverse=True
        )
        self.current_symbol = prediction[0][0]
        if self.current_symbol == "blank":
            for i in ''.join([str(digit) for digit in range(10)] + list(string.ascii_uppercase)): #changes
done
                self.ct[i] = 0
        self.ct[self.current_symbol] += 1
        if self.ct[self.current_symbol] > 20:
            for i in ''.join([str(digit) for digit in range(10)] + list(string.ascii_uppercase)): #changes
done
                if i == self.current_symbol:
                    continue
                tmp = self.ct[self.current_symbol] - self.ct[i]
                if tmp < 0:
                    tmp *= -1
                if tmp <= 20:
                    self.ct["blank"] = 0
                    for i in ''.join([str(digit) for digit in range(10)] + list(string.ascii_uppercase)):
#changes done
                        self.ct[i] = 0
                    return
            self.ct["blank"] = 0
```

```
        for i in ''.join([str(digit) for digit in range(10)] + list(string.ascii_uppercase)): #changes
done
            self.ct[i] = 0
        if self.current_symbol == "blank":
            if self.blank_flag == 0:
                self.blank_flag = 1
                if len(self.str) > 0:
                    self.str += " "
                self.str += self.word
                self.word = ""
        else:
            if len(self.str) > 16:
                self.str = ""
            self.blank_flag = 0
            self.word += self.current_symbol

    def destructor(self):
        print("Closing Application...")
        self.root.destroy()
        self.vs.release()
        cv2.destroyAllWindows()

    def destructor1(self):
        print("Closing Application...")
        self.root1.destroy()


print("Starting Application...")
pba = Application()
pba.root.mainloop()
```

SCREENSHOTS

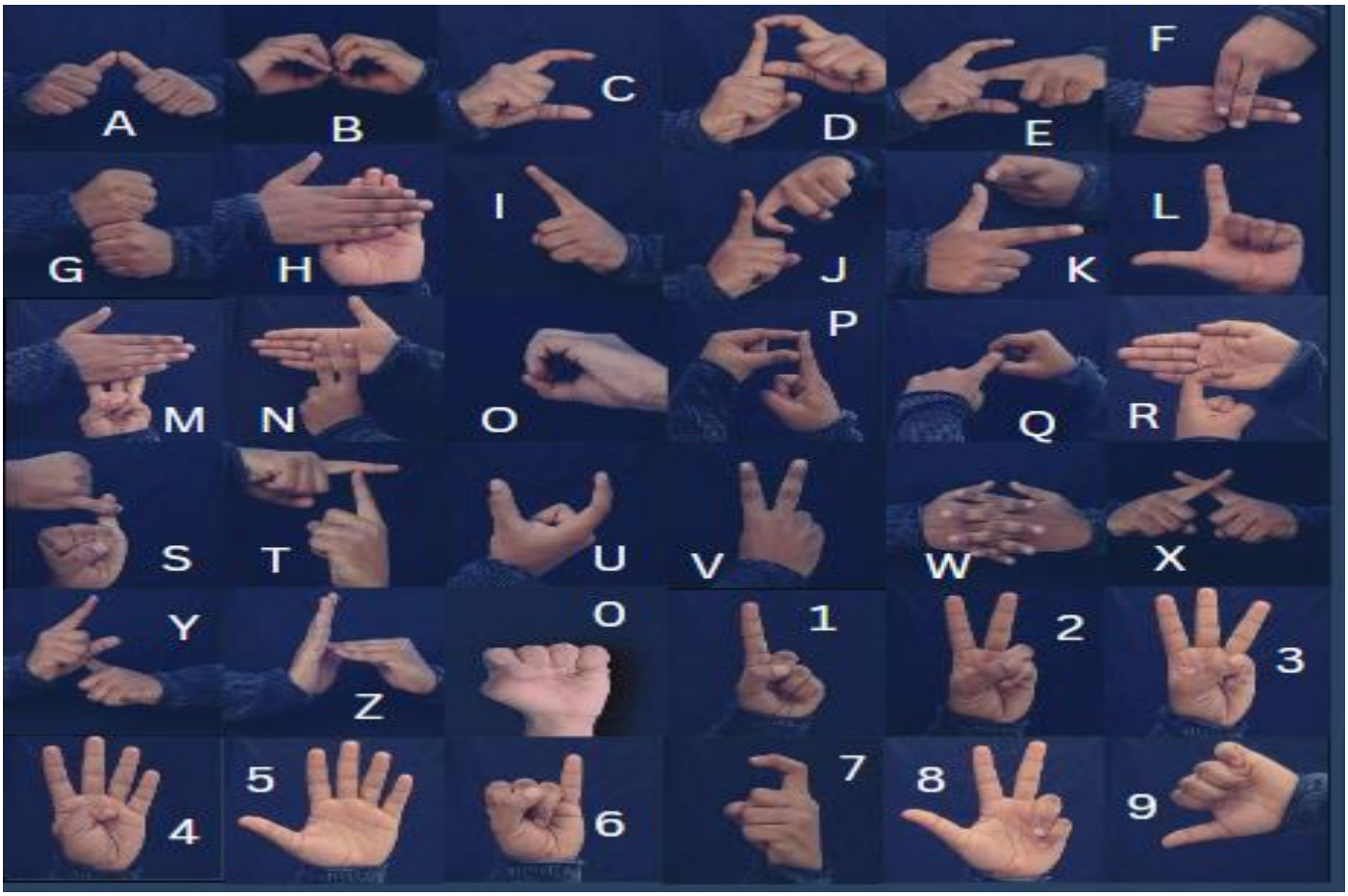
Proposed CNN:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 64)      640

 max_pooling2d (MaxPooling2D  (None, 63, 63, 64)       0
 )

 conv2d_1 (Conv2D)           (None, 61, 61, 64)        36928

 max_pooling2d_1 (MaxPooling  (None, 30, 30, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 28, 28, 64)        36928

 max_pooling2d_2 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 flatten (Flatten)           (None, 12544)             0

 dense (Dense)               (None, 128)               1605760

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 36)                4644

=================================================================
Total params: 1,684,900
Trainable params: 1,684,900
Non-trainable params: 0
```

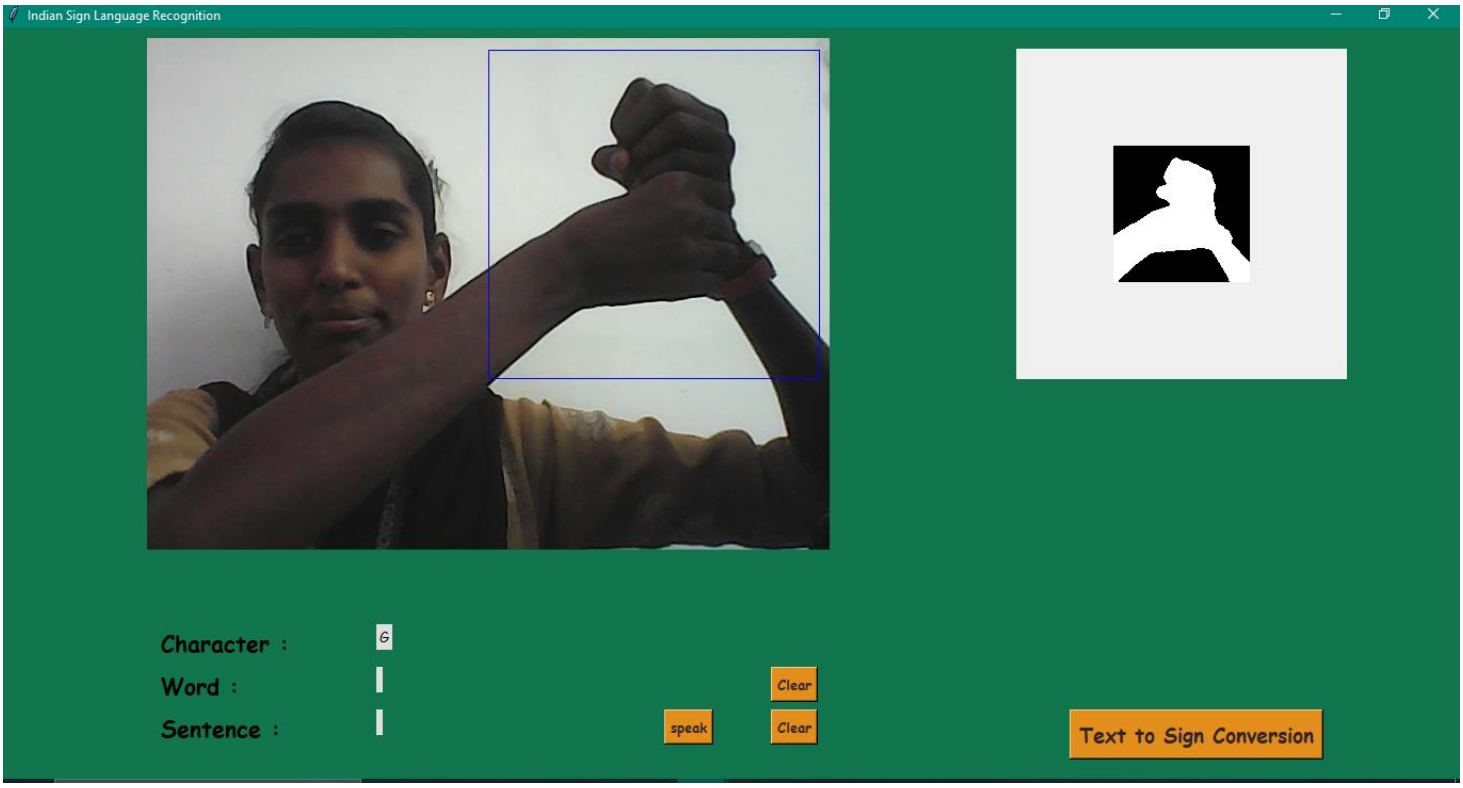
Epoch:

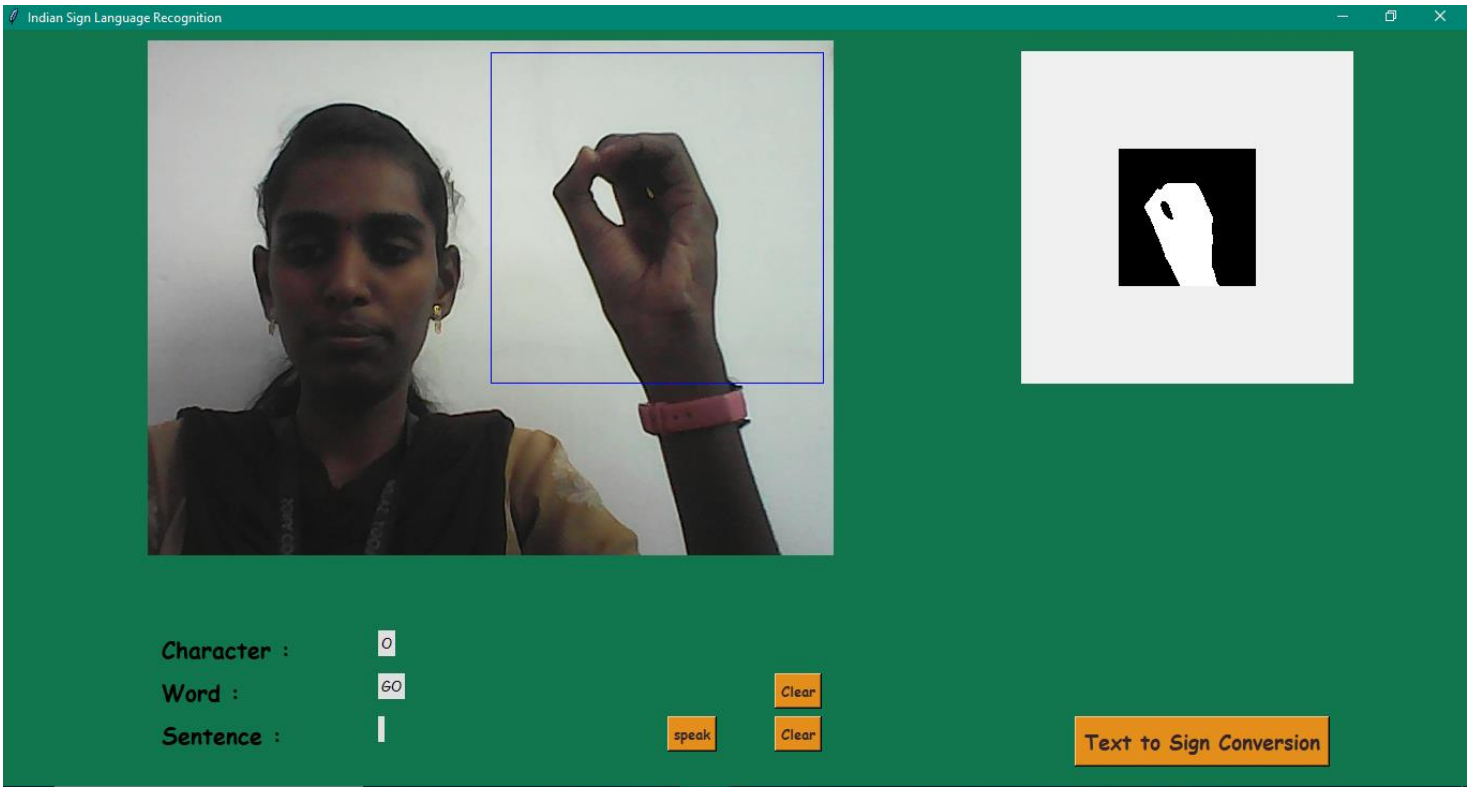
```
Epoch 1/5
5040/5040 [==============================] - 432s 85ms/step - loss: 1.6404 - accuracy: 0.5292 - val_loss: 0.2879 - val_accurac
y: 0.9181
Epoch 2/5
5040/5040 [==============================] - 392s 78ms/step - loss: 0.5592 - accuracy: 0.8214 - val_loss: 0.1744 - val_accurac
y: 0.9588
Epoch 3/5
5040/5040 [==============================] - 410s 81ms/step - loss: 0.3742 - accuracy: 0.8802 - val_loss: 0.0849 - val_accurac
y: 0.9782
Epoch 4/5
5040/5040 [==============================] - 450s 89ms/step - loss: 0.2726 - accuracy: 0.9109 - val_loss: 0.0536 - val_accurac
y: 0.9898
Epoch 5/5
5040/5040 [==============================] - 446s 89ms/step - loss: 0.2198 - accuracy: 0.9272 - val_loss: 0.0728 - val_accurac
y: 0.9838
```

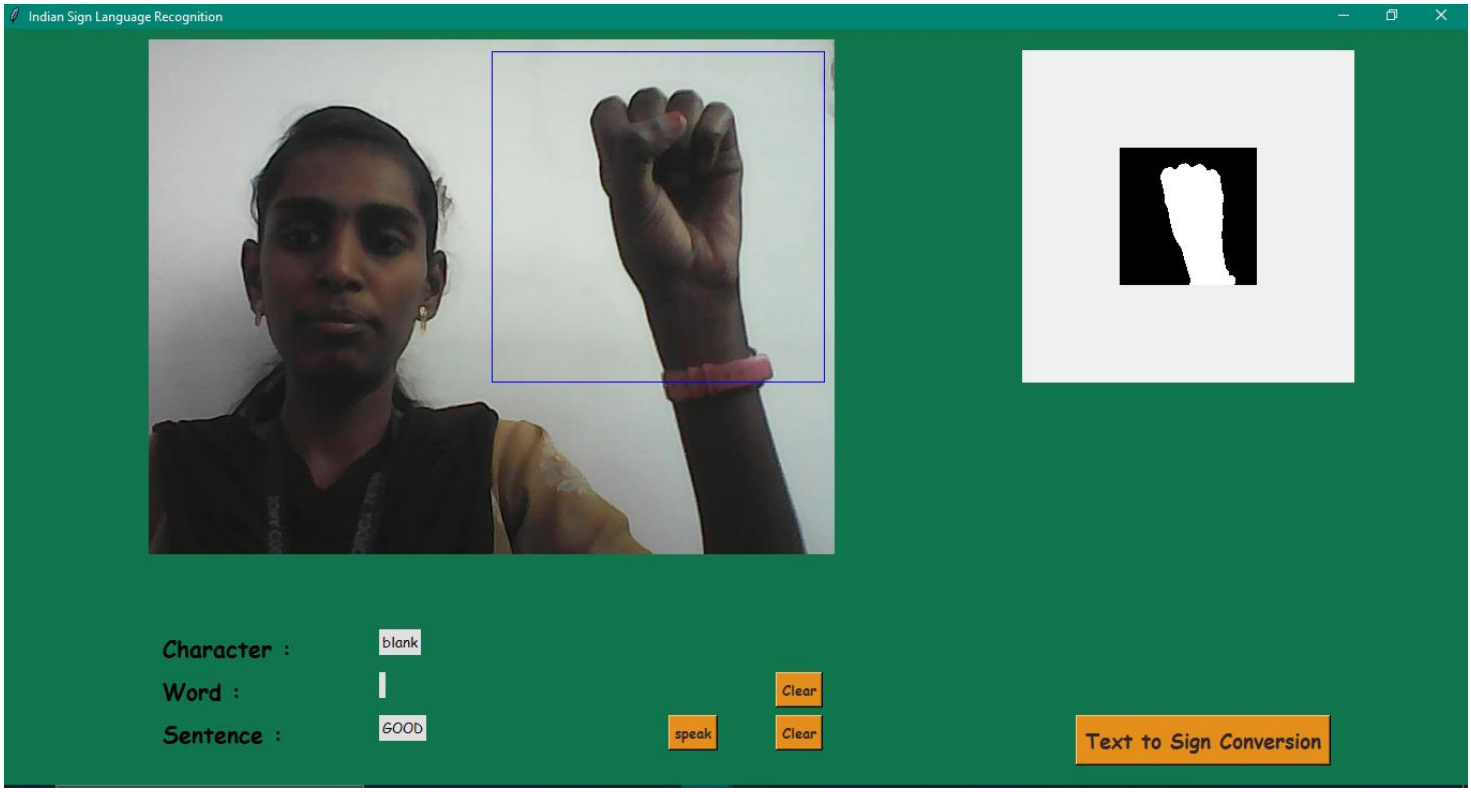
Hand Gestures:

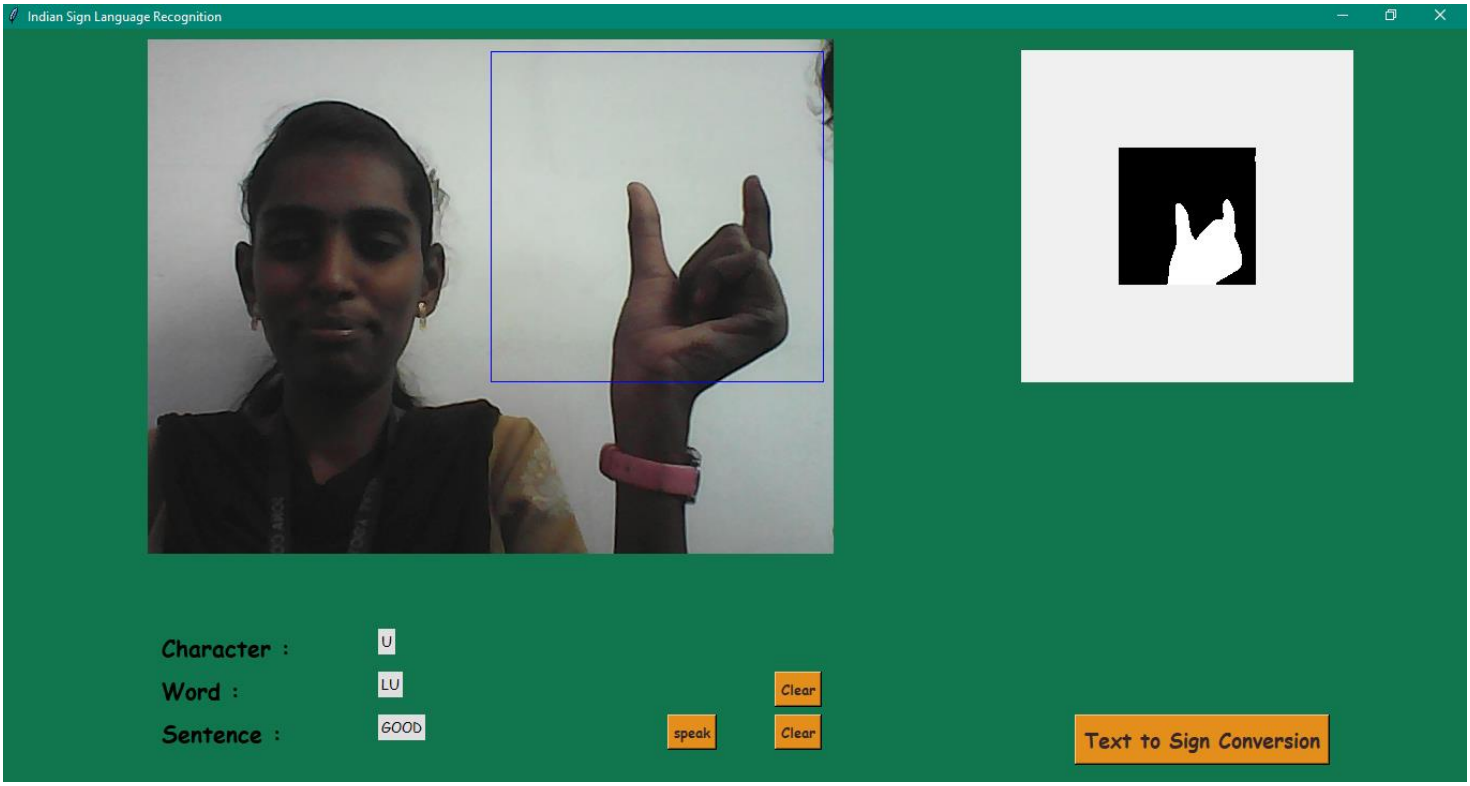


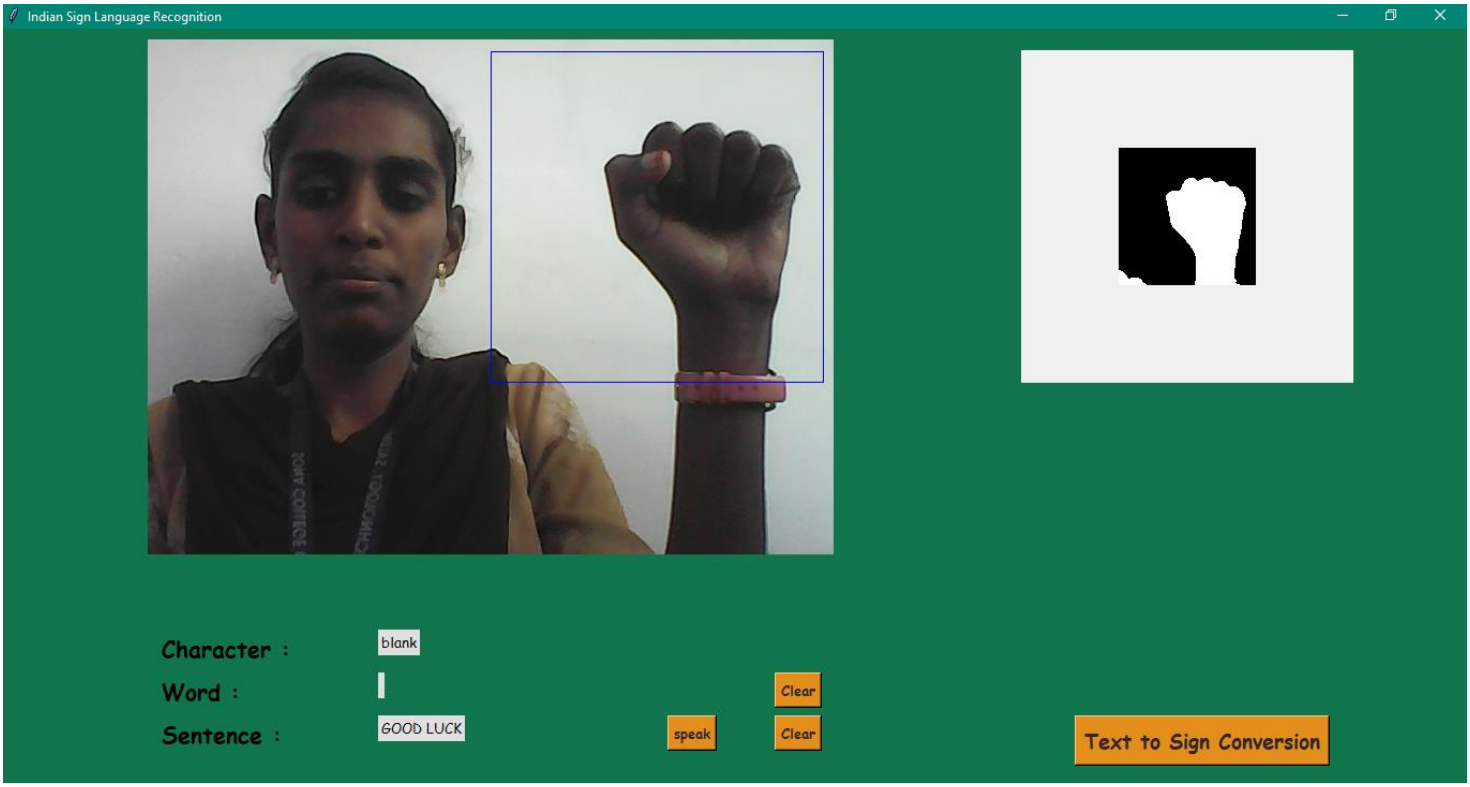
Real time prediction:

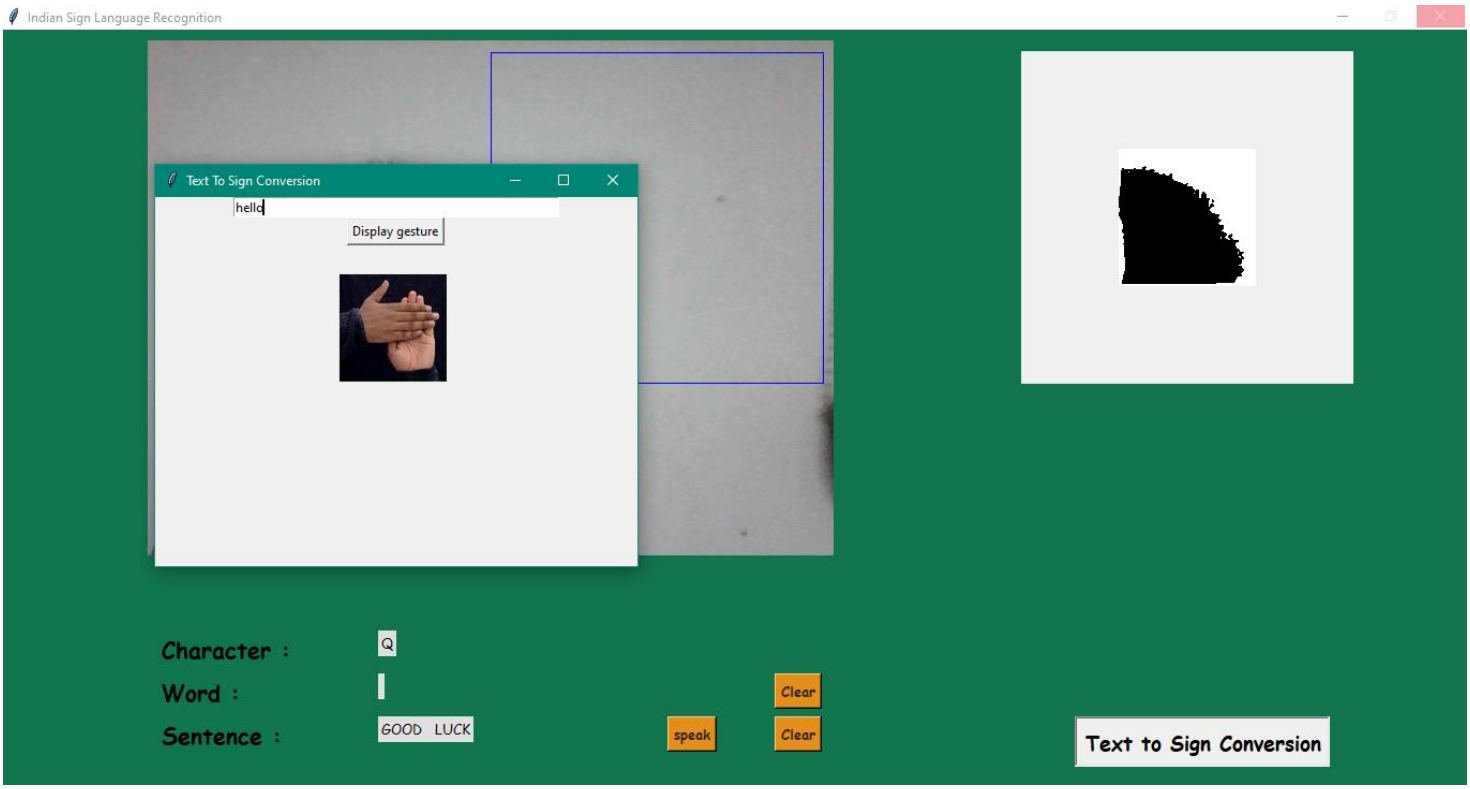





Indian Sign Language Recognition
Character :
blank
Word :
Sentence :
GOOD
speak
Clear
Clear
Text to Sign Conversion


Indian Sign Language Recognition
Character :
U
Word :
LU
Sentence :
GOOD
speak
Clear
Clear
Text to Sign Conversion


Indian Sign Language Recognition
Character :
blank
Word :
Sentence :
GOOD LUCK
speak
Clear
Clear
Text to Sign Conversion


Indian Sign Language Recognition
Text To Sign Conversion
hello
Display gesture
Character :
Q
Word :
Sentence :
GOOD LUCK
speak
Clear
Clear
Text to Sign Conversion

REFERENCES

[1] Triwijoyo, Bambang & Karnaen, Ahmat & Adil, Ahmat. (2023). Deep Learning Approach for Sign Language Recognition. 9. 12-21. 10.26555/jiteki.v9i1.25051

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

[3] Abey Abraham, V Rohini,Real time conversion of sign language to speech and prediction of gestures using Artificial Neural Network,Procedia Computer Science,Volume 143,2018,Pages 587-594,ISSN1877-0509,https://doi.org/10.1016/j.procs.2018.10.435.

[4] NB, M.K., 2018. Conversion of sign language into text. International Journal of Applied Engineering Research, 13(9), pp.7154-7161.

[5] Mehreen Hurroo , Mohammad Elham, 2020, Sign Language Recognition System using Convolutional Neural Network and Computer Vision, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 09, Issue 12 (December 2020)

[6] KASAPBAŞI, Ahmed & Elbushra, Ahmed & AL-HARDANEE, Omar & YILMAZ, Arif. (2022). DeepASLR: A CNN based Human Computer Interface for American Sign Language Recognition for Hearing-Impaired Individuals. Computer Methods and Programs in Biomedicine Update. 2. 100048. 10.1016/j.cmpbup.2021.100048

[7] Thakur, Amrita & Budhathoki, Pujan & Upreti, Sarmila & Shrestha, Shirish & Shakya, Subarna. (2020). Real Time Sign Language Recognition and Speech Generation. Journal of Innovative Image Processing. 2. 65-76. 10.36548/jiip.2020.2.001

[8] Rachana Patil, Vivek Patil, Abhishek Bahuguna, Gaurav Datkhile, Indian Sign Language Recognition using Convolutional Neural Network, ITM Web Conf. 40 03004 (2021),DOI: 10.1051/itmconf/20214003004

[9] Kodandaram, Satwik Ram & Kumar, N. & Gl, Sunil. (2021). Sign Language Recognition. Turkish Journal of Computer and Mathematics Education (TURCOMAT). 12. 994-1009

[10] Singh, D.K., 2021. 3d-cnn based dynamic gesture recognition for indian sign language modeling. Procedia Computer Science, 189, pp.76-83

[11] Ghaleb, F. , Youness, E. , Elmezain, M. and Dewdar, F. (2015) Vision-Based Hand Gesture Spotting and Recognition Using CRF and SVM. Journal of Software Engineering and Applications, 8, 313-323. doi: 10.4236/jsea.2015.87032

[12] Srivastava, Sharvani & Gangwar, Amisha & Mishra, Richa & Singh, Sudhakar. (2022). Sign Language Recognition System Using TensorFlow Object Detection API. 10.1007/978-3-030-96040-7_48

[13] Huang, J., Zhou, W., Zhang, Q., Li, H., and Li, W., "Video-based Sign Language Recognition without Temporal Segmentation", 2018. doi:10.48550/arXiv.1801.10111.

[14] Sreenivas, Arvind & Maheshwari, Mudit & Jain, Saiyam & Choudhary, Shalini & G, Dr. Vadivu. (2020). Indian Sign Language Communicator Using Convolutional Neural Network. International Journal of Advanced Science and Technology. 29. 11015-11031.

[15] Ahuja, M.K. and Singh, A., 2015. Hand gesture recognition using PCA. International Journal of Computer Science Engineering and Technology (IJCSET), 5(7), pp.267-27.

[16] Fang, G., Gao, W., & Zhao, D. (2003). Large vocabulary sign language recognition based on hierarchical decision trees. Proceedings of the 5th International Conference on Multimodal Interfaces - ICMI '03. doi:10.1145/958432.958458

[17] Kumar, A., Thankachan, K. and Dominic, M.M., 2016, March. Sign language recognition. In 2016 3rd international conference on recent advances in information technology (RAIT) (pp. 422-428). IEEE.

[18] Pankajakshan, P. C., & Thilagavathi B. (2015). Sign language recognition system. 2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS). doi:10.1109/iciiecs.2015.7192910

[19] Sun, Jing-Hao; Ji, Ting-Ting; Zhang, Shu-Bin; Yang, Jia-Kui; Ji, Guang-Rong (2018). [IEEE 2018 12th International Symposium on Antennas, Propagation and EM Theory (ISAPE) - Hangzhou, China (2018.12.3-2018.12.6)] 2018 12th International Symposium on Antennas, Propagation and EM Theory (ISAPE) - Research on the Hand Gesture Recognition Based on Deep Learning., (), 1–4. doi:10.1109/ISAPE.2018.8634348

[20] Huang, Hanwen & Chong, Yan-Wen & Nie, Congchong & Pan, Shaoming. (2019). Hand Gesture Recognition with Skin Detection and Deep Learning Method. Journal of Physics: Conference Series. 1213. 022001. 10.1088/1742-6596/1213/2/022001.

[21] Nikam, A.S. and Ambekar, A.G., 2016, November. Sign language recognition using image-based hand gesture recognition techniques. In 2016 online international conference on green engineering and technologies (IC-GET) (pp. 1-5). IEEE.

[22] Karayilan, Tulay; Kilic, Ozkan (2017). [IEEE 2017 International Conference on Computer Science and Engineering (UBMK) - Antalya (2017.10.5-2017.10.8)] 2017 International Conference on Computer Science and Engineering (UBMK) - Sign language recognition. , (), 1122–1126. doi:10.1109/UBMK.2017.8093509

[23] Mr. G. Sekhar Reddy, A. Sahithi , P. Harsha Vardhan , P. Ushasri, Conversion of Sign Language Video to Text and Speech,https://doi.org/10.22214/ijraset.2022.42078

[24] Citra Suardi;Anik Nur Handayani;Rosa Andrie Asmara;Aji Prasetya Wibawa;Lilis Nur Hayati;Huzain Azis; (2021). Design of Sign Language Recognition Using E-CNN. 2021 3rd East Indonesia Conference on Computer and Information Technology (EIConCIT), (), –. doi:10.1109/eiconcit50028.2021.9431877

[25] Bhagat, Neel Kamal; Vishnusai, Y.; Rathna, G. N. (2019). [IEEE 2019 Digital Image Computing: Techniques and Applications (DICTA) - Perth, Australia (2019.12.2-2019.12.4)] 2019 Digital Image Computing: Techniques and Applications (DICTA) - Indian Sign Language Gesture Recognition using Image Processing and Deep Learning. , (), 1–8. doi:10.1109/DICTA47822.2019.8945850

[26] C J, Sruthi; A, Lijiya (2019). [IEEE 2019 International Conference on Communication and Signal Processing (ICCSP) - Chennai, India (2019.4.4-2019.4.6)] 2019 International Conference on Communication and Signal Processing (ICCSP) - Signet: A Deep Learning based Indian Sign Language Recognition System., (), 0596–0600. doi:10.1109/ICCSP.2019.8698006

[27] Ghaleb, F.F.M., Youness, E.A., Elmezain, M. and Dewdar, F.S. (2015) Vision-Based Hand Gesture Spotting and Recognition Using CRF and SVM. Journal of Software Engineering and Applications, 8, 313-323. http://dx.doi.org/10.4236/jsea.2015.87032

[28] Shirbhate, R.S., Shinde, V.D., Metkari, S.A., Borkar, P.U. and Khandge, M.A., 2020. Sign language recognition using machine learning algorithm. International Research Journal of Engineering and Technology (IRJET), 7(03), pp.2122-2125.

[29] Sarkar, A., Talukdar, A.K. and Sarma, K.K., 2020. CNN-based real-time Indian sign language recognition system. In Advances in Computational Intelligence and Informatics: Proceedings of ICACII 2019 (pp. 71-79). Springer Singapore.

[30] Sharma, A., Sharma, N., Saxena, Y. et al. Benchmarking deep neural network approaches for Indian Sign Language recognition. Neural Comput & Applic 33, 6685–6696 (2021). https://doi.org/10.1007/s00521-020-05448-8

[31] Utaminingrum, Fitri et al. "Alphabet Sign Language Recognition Using K-Nearest Neighbor Optimization." *J. Comput.* 14 (2019): 63-70.

[32] Dewinta, & Heryadi, Yaya. (2015). American Sign Language-Based Finger-spelling Recognition using k-Nearest Neighbours Classifier. 10.1109/ICoICT.2015.7231481.

[33] Das, S., Imtiaz, M.S., Neom, N.H., Siddique, N. and Wang, H., 2023. A hybrid approach for Bangla sign language recognition using deep transfer learning model with random forest classifier. Expert Systems with Applications, 213, p.118914.