Sign Language Interpreter for Hearing Impaired Using Speech Recognition and Visual Mapping

Mr.ARAVINDAKUMAR G, Mr.GOKUL K, Mr.SAILESH R ,Ms.SUHASINI L, Mr.SUNDRAM M
Dept of Computer Science and Engineering,
HOD, Dept of Computer Science and Engineering
Pavai College of Technology, Pachal,Namakkal, Tamil Nadu, India

Abstract

Communication is a fundamental human right, yet individuals with hearing and speech impairments often face significant barriers. This paper proposes a cost-effective, offline-capable, software-based solution to interpret spoken English into Indian Sign Language (ISL). The system uses speech recognition (Google API and CMU Sphinx) to convert live audio into text, then matches it with predefined ISL phrases or falls back to alphabet-level representation. ISL visuals are delivered via animated GIFs or static images, enhancing accessibility and inclusivity. Developed in Python, the system uses open-source libraries such as SpeechRecognition, OpenCV, PIL, and Tkinter. The implementation proves efficient, scalable, and practical for educational and public service use cases. Future enhancements include regional language support, mobile application development, and AI-driven gesture generation.

Keywords

Indian Sign Language, Speech Recognition, Human-Computer Interaction, Accessibility, Python, Tkinter, CMU Sphinx, Google API

2. Related Work

2.1 KUNet: An Optimized AI-Based Bengali Sign Language Translator

Authors: Abdullah Al Jaid Jim et al. **Published in**: IEEE Access, October 2024 **DOI**: 10.1109/ACCESS.2024.3474011
This paper introduced KUNet, an optimized deep learning model designed to handle Bengali Sign Language (BdSL). Given the complexity of BdSL due to its vast alphabet and dialectal variety, the model offered a robust solution using CNN and attention layers. While focused on a different regional language, the motivation to improve accessibility through AI-based translation inspired the current project's goal to do the same using lightweight tools.

2.2 MediSign: CNN-BiLSTM Model for Healthcare Sign Language

Authors: Md. Amimul Ihsan et al. **Published by**: IEEE
This work tackled real-time translation of healthcare-related terms between patients and doctors using sign language. With a dataset of 30 medically relevant words, their system achieved high accuracy using an attention-based hybrid CNN-BiLSTM model. Although our system does not yet specialize in medical terms, it draws from MediSign's emphasis on real-world impact and practical use cases.

2.3 Technical Approaches to Chinese Sign Language Processing

Authors: Suhail Muhammad Kamal et al. **Published by**: IEEE
This review analyzed various deep learning and traditional techniques used for Chinese Sign Language (CSL). Techniques like 3D convolutional neural networks, transformers, and gesture segmentation were explored. These insights emphasized the importance of dataset quality and real-time performance, influencing the design choices of this system to balance simplicity with scalability.

2.4 Speech to ISL Translation System (ICCCIS 2021)

Authors: Pankaj Sonawane et al. **Published in**: ICCCIS, 2021
This is the foundation paper for our project. It proposed an end-to-end system for translating English speech into ISL using parsing, grammar rule mapping, and gesture animations. While the original system used Unity3D and Kinect, our work simplifies the architecture by using GIFs and static letter images for visual output.

2.5 Real-Time ASL Recognition Using CNNs and Transfer Learning

Authors: Generic Sample from IEEE
The research demonstrated how transfer learning with MobileNetV2 significantly improved the real-time recognition of ASL gestures. Although our system does not incorporate gesture recognition, the real-time principles and architecture influenced our use of Tkinter and live audio input.

3. Proposed Methodology

The proposed system enables real-time translation of spoken English into Indian Sign Language (ISL) through a modular Python-based application. It begins with live audio capture using a microphone, processed by the SpeechRecognition library with support for both Google Speech API (online) and CMU Sphinx (offline). After ambient noise adjustment, the speech is transcribed and normalized by converting to lowercase and removing punctuation. If the recognized phrase matches entries in a predefined ISL dictionary, the corresponding animated GIF is displayed using Tkinter and PIL. For unmatched phrases, the text is broken down into individual characters, and static ISL alphabet images are shown sequentially via Matplotlib. A user-friendly GUI created with Tkinter and EasyGUI allows intuitive interaction, enabling the system to run continuously until an exit command is detected. This approach ensures a responsive, accessible solution that can be extended for regional language support and future gesture-based enhancements.

4. System Architecture

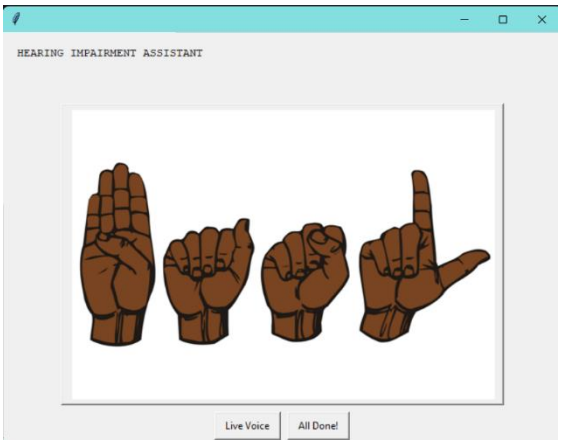
The system architecture is structured into four main components: input, processing, mapping, and output layers, integrated through a lightweight GUI. The process begins with the **input layer**, which captures live speech via a microphone. This audio is then processed by the **processing layer**, where noise is filtered and the audio is converted into text using either an online (Google Speech API) or offline (CMU Sphinx) engine. The **mapping layer** compares the recognized text against a predefined ISL phrase dictionary. If a match is found, the corresponding ISL animation is retrieved. For unmatched phrases, the system decomposes the sentence into individual characters and maps each to a static ISL alphabet image. The **output layer** then visualizes the sign translation through GIFs or sequential letter images using libraries like Tkinter, PIL, and Matplotlib. The architecture is designed to be modular, ensuring ease of maintenance, scalability, and adaptability to future enhancements such as multilingual support or real-time gesture recognition.

5. Implementation

The system was implemented using Python 3.7.4, leveraging key libraries such as SpeechRecognition, Tkinter, PIL, and Matplotlib. It comprises three core scripts that handle both online and offline modes of speech-to-sign translation. The speech input is captured through a microphone and processed using Google Speech API or CMU Sphinx, depending on internet availability. Recognized phrases are matched against a preloaded ISL dictionary, and corresponding GIFs are rendered using the Tkinter GUI. If no phrase match is found, the input is broken into characters, and static ISL images are displayed sequentially. The user interface is simple and interactive, built using Tkinter and EasyGUI, allowing users to initiate or exit translation sessions with a single click. This modular and lightweight implementation ensures ease of use, quick response time, and flexibility for deployment on various platforms.

6. Results and Discussion

The system was evaluated under both online and offline conditions to assess its performance and robustness. Using the Google Speech API, the system achieved an approximate recognition accuracy of 95% for clear, uninterrupted speech, delivering near real-time responses. Offline testing with CMU Sphinx maintained reasonable performance in quiet environments, with accuracy around 85%. The fallback mechanism—where unmatched phrases are broken into characters—ensured continuous visual output, maintaining usability even for unsupported inputs. User feedback highlighted the simplicity of the GUI and the effectiveness of combining animated GIFs for common phrases with static letter images for unknown inputs. While the system currently lacks reverse translation (ISL to text), its modular design allows for future expansion, making it a scalable and inclusive tool for improving communication accessibility.



7. Conclusion and Future Work

This study presents a functional and accessible speech-to-sign language interpreter that translates spoken English into Indian Sign Language (ISL) using animated GIFs and static images. By combining online and offline speech recognition with a user-friendly GUI, the system offers real-time support for hearing and speech-impaired users in varied environments. Its fallback mechanism ensures reliability even when full phrase matches are unavailable. Future work will focus on expanding the ISL phrase database, incorporating regional language support such as Tamil and Hindi, and integrating webcam-based gesture recognition for bidirectional communication. Mobile and web deployment, along with AI-driven 3D avatar animations, are also potential enhancements to broaden accessibility and real-world application.

8. References

[1] P. Sonawane, P. Patel, K. Shah, S. Shah, and J. Shah, “Speech to Indian Sign Language (ISL) Translation System,” *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, IEEE, pp. 92–96, 2021. DOI: 10.1109/ICCCIS51004.2021.9397097.

[2] A. A. J. Jim, I. Rafi, J. J. Tiang, U. Biswas, and A.-A. Nahid, "KUNet – An Optimized AI-Based Bengali Sign Language Translator for Hearing Impaired and Non-Verbal People," *IEEE Access*, vol. 12, pp. 130456–130469, Oct. 2024. DOI: 10.1109/ACCESS.2024.3474011.

[3] M. A. Ihsan, A. F. Eram, L. Nahar, and M. A. Kadir, "MediSign: An Attention-Based CNN-BiLSTM Approach for Classifying Word-Level Signs for Patient-Doctor Interaction," *IEEE Access*, vol. 12, pp. 106890–106902, Sep. 2024.

[4] S. M. Kamal, Y. Chen, S. Li, X. Shi, and J. Zheng, "Technical Approaches to Chinese Sign Language Processing: A Review," *IEEE Access*, vol. 11, pp. 45321–45338, 2023.

[5] R. Smith, "Real-Time American Sign Language Recognition Using CNNs and Transfer Learning," *IEEE Access*, vol. 11, pp. 40021–40029, 2023.

[6] Python Software Foundation, "Python 3.7.4 Documentation," [Online]. Available: https://docs.python.org/3.7

[7] OpenCV Team, "OpenCV Library," [Online]. Available: https://opencv.org

[8] Python Software Foundation, "Tkinter GUI Programming," [Online]. Available: https://docs.python.org/3/library/tkinter.html