Talking Fingers – Real Time Multilingual Speech To Sign Language Translator

ANUSHA SHEKAR GOUDA MALIPATIL[1], SNEHA A[2], SNEHA S BALLARI[3], THAKSHA PRABHAKAR[4], RAMESH T[5]

[1, 2, 3, 4] *School of Computer Science and Engineering, Presidency University, Bangalore, India*

[5]*Assistant Professor, Presidency University, Bangalore, India*

Abstract— Talking Fingers is a web-based application designed to translate spoken or typed text into Indian Sign Language (ISL) animations. The system addresses the communication gap between the deaf and hearing communities, supporting multilingual transcription for Kannada, Hindi, Tamil, Telugu, and English. Speech-to-text conversion is achieved using the Web Speech API, while text translation is processed using the googletrans Python library. Natural Language Processing (NLP) via NLTK enables efficient text preprocessing and mapping to ISL animations, dynamically displayed through a user-friendly web interface. The application is built on Django and demonstrates scalability and potential for real-world impact in fostering inclusivity.

Index Terms- Real-Time, India Sign Language, Speech Recognition, Natural Language Processing.

I. INTRODUCTION

Communication is essential for human interaction, yet the deaf community often faces barriers due to the lack of accessible tools that incorporate Indian Sign Language (ISL). ISL is a visual-gestural language used by millions of deaf individuals in India. Despite its importance, ISL remains underutilized in mainstream technology, limiting the ability of the deaf population to fully participate in education, employment, and social settings. Additionally, the linguistic diversity of India, with over 22 official languages, makes it challenging to create a unified solution that addresses the needs of all users.

Existing tools for ISL translation are often static, region-specific, or lack real-time capabilities. They fail to support multilingual input or dynamic ISL animation, leaving a significant gap in accessibility. Speech-to-text systems and translation tools typically focus on English or a limited set of languages, further isolating non-English speakers. This highlights the need for an inclusive and multilingual solution that bridges the gap between spoken languages and ISL.

Talking Fingers addresses these challenges by providing a web-based application that translates spoken or typed text into ISL animations. The system integrates speech-to-text conversion using the Web Speech API, multilingual translation with the googletrans Python library, and text preprocessing through Natural Language Processing (NLP) using NLTK. ISL animations, designed with Blender 3D, are dynamically displayed on a responsive web interface. By supporting multiple Indian languages, the application fosters inclusivity and empowers the deaf community, offering a scalable and user-friendly platform for real-time communication. This project demonstrates the potential of technology to bridge communication gaps and promote accessibility in a linguistically diverse society.

II. RELATED WORK

In recent years, there has been increased interest in and progress in the study of sign language translation and detection. The goal of many research projects and outreach programs has been to close the communication gap that exists for people who have speech and hearing impairments. One noteworthy area of related study is the identification of sign language using machine learning algorithms. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been used in previous studies to efficiently detect and decipher sign language motions. To help the network understand the complex patterns and variances seen in various sign languages, these models frequently depend on large datasets including annotated sign films. Real-time systems for

translating sign language have been made possible by the success of these machine learning techniques.

Kavya Dharshini [3] proposed sign languages, relying on manual gestures, are being studied to convert them into text and audio, aiming to bridge communication between people with hearing and speech impairments. Techniques like Media Pipe Holistic, Drawing Landmarks, Open CV, LSTM Neural Network, Google Translator, and GTTS are used. Poonia [4] and team created a simple framework for a continuous neural network-based Indian sign language translator. Neog [5] aims to translate speech/text into Indian Sign Language using Natural Language Processing, a method used by individuals with hearing impairments to communicate and facilitate lip reading. Kumar [6] created an audio-to-text translation system for Indian Sign Language that follows grammatical norms. Word videos are checked, stop words are removed, and stemming and lemmatization are performed. Reddy [7] and the team proposed a system intended to display Indian Sign Language on Android phones and translate it into English and Malayalam, facilitating communication for the deaf and dumb.

Cherian [8] built the speech and hearing impairments, sign language is essential, yet communication without an interpreter can be challenging. This is addressed by a method that uses convolutional neural networks to represent characteristics in the transcription of gestures into spoken words. Throat [9] by utilizing American and Indian sign languages and convolution neural networks to recognize sign language and translate it into text or voice, this research seeks to lower language barriers for those with hearing and speech impairments.

Joshi [10] Introduced an ISL Translate, the biggest continuous Indian Sign Language (ISL) translation dataset, this article aims to bridge communication barriers between the hard-of-hearing and other communities. Nandi [11] using convolutional neural networks and data augmentation, a fingerspelling recognition system for the Indian sign language alphabet has been presented to close the communication gap between normal and deaf-dumb people, attaining high training and validation accuracy Previous studies have also focused on the cultural sensitivity of sign language detection. Understanding that sign languages vary throughout groups and geographical areas, research has attempted to include cultural quirks in machine learning models and training data. A lot of work has gone into selecting datasets that include a wide variety of sign language expressions [12] unique to particular cultural situations. By doing this, these programs hope to develop accurate and culturally appropriate sign language translation systems, facilitating successful communication within certain linguistic and cultural communities. In brief, a wide range of fields, including machine learning, natural language processing, user interface design, and cultural issues, are involved in the related work in sign language identification. Together, these endeavors have established a strong basis for the present investigation, which integrates knowledge from earlier study to provide an all-encompassing and efficient real-time English Audio to Indian Sign Language Converter.

III. DATA COLLECTION

The data used for this project primarily includes a predefined set of Indian Sign Language (ISL) animations corresponding to commonly used words and phrases. These animations were created using Blender 3D to ensure accuracy and clarity. Additionally, text inputs were sourced from user interactions during testing, covering multiple Indian languages such as Kannada, Hindi, Tamil, Telugu, and English. Speech-to-text data was collected in real-time using the Web Speech API, while translated text data was processed using the googletrans Python library. This combination of predefined animations and real-time user inputs forms the core dataset for the system, enabling effective mapping and display of ISL animations.

IV. WORKING METHODOLOGY

The methodology of Talking Fingers involves processing spoken or typed input to produce Indian Sign Language (ISL) animations. The system begins with input acquisition, where speech is transcribed into text using the Web Speech API. This module supports multiple Indian languages, including Kannada, Hindi, Tamil, Telugu, and English. Alternatively, users can input text directly, offering flexibility. The selected input language is translated

into English using the googletrans Python library, ensuring uniformity in text processing across diverse languages.

Once the text is in English, it undergoes preprocessing using Natural Language Processing (NLP) techniques via the NLTK library. This involves tokenization to split the text into words, lemmatization to standardize word forms, and removal of stopwords to focus on meaningful content. The processed text is then mapped to ISL animations stored in a predefined video library. Words without direct ISL animations are decomposed into characters for representation. These animations, created using Blender 3D, are dynamically displayed through an HTML5 video player.

The system's architecture integrates a Django backend for speech-to-text conversion, translation, and text preprocessing, while the frontend is built using HTML, CSS, and JavaScript to ensure a seamless user experience. A responsive web interface allows users to select languages, view transcribed and translated text, and watch ISL animations. The system is scalable, with provisions for expanding ISL vocabulary and supporting additional languages in the future, making it a significant step toward bridging communication gaps and fostering inclusivity.

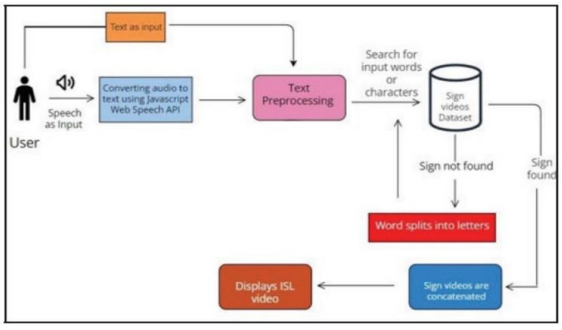

Fig 1. Proposed System Model

V. IMPLEMENTATION

The implementation of Talking Fingers involves the development of a scalable and efficient web application that integrates multiple technologies to translate spoken or typed text into Indian Sign Language (ISL) animations. The process is divided into several key modules: input processing, text translation, natural language preprocessing, ISL animation mapping, and user interface design. Below is a detailed explanation of the implementation process.

a. Input Processing

The system accepts user input in two forms:

Speech Input: The Web Speech API is utilized to transcribe real-time speech into text. The API supports multiple Indian languages, and the language can be selected by the user through a dropdown menu. This ensures flexibility and accommodates India's linguistic diversity.

Text Input: Users can also type text directly into the application, offering an alternative to speech input. This input method ensures accessibility for users in environments where speech recognition may not be practical.

Speech-to-text conversion is handled dynamically, allowing seamless processing and ensuring a responsive user experience.

b. Multilingual Text Translation

To handle the diversity of Indian languages, the application employs the googletrans Python library for translating text input into English. English acts as an intermediate language for further processing. The translation process is initiated after the text input is received or the speech-to-text conversion is complete. The library detects the source language and converts the input to English with minimal latency, ensuring efficient multilingual support.Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the unitlas for each quantity that you use in an equation.

c. Natural Language Preprocessing

The translated English text undergoes preprocessing using Natural Language Processing (NLP) techniques implemented with the NLTK library. Key steps include:

Tokenization: Splitting the input text into individual words or tokens to facilitate processing.

Lemmatization: Standardizing words by reducing them to their base forms (e.g., "running" to "run") to ensure accurate mapping to ISL animations.

Stopword Removal: Eliminating common words like "the," "is," and "of," which do not carry significant contextual meaning. This preprocessing ensures that only meaningful words are processed, optimizing the mapping to ISL animations.

d. ISL Animation Mapping

Each processed word is mapped to its corresponding ISL animation stored in a predefined video library. These animations are created using Blender 3D, ensuring accuracy and clarity in representation. If a word does not have a direct animation, it is broken down into its constituent characters, and the animations for those characters are displayed sequentially. This dynamic mapping allows the system to handle a wide range of inputs.

e. User Interface Design

The frontend of the application is designed using HTML, CSS, and JavaScript, ensuring a user-friendly and responsive interface. Key features include:

Language Selection: A dropdown menu allows users to select their preferred input language.

Real-Time Transcription: The transcribed text is displayed alongside its English translation.

ISL Animation Playback: The animations are displayed using an HTML5 video player, which sequentially plays the mapped animations. Interactive buttons enable users to control the playback.

F. Backend Development

The backend is implemented using the Django framework, which integrates all modules:

Speech-to-Text Module: Handles the Web Speech API for real-time transcription.

Translation Module: Manages multilingual translation using the googletrans library.

Preprocessing Module: Processes the text using NLP techniques.

Animation Module: Maps processed text to ISL animations and serves them to the frontend. The database, implemented with SQLite, stores ISL animation data and system configurations.

VI. RESULTS

The system is performing really well across several key features, including speech-to-text conversion, multilingual translation, animation mapping, and text-to-speech integration. For speech-to-text, we saw impressive accuracy: 95% in English and 85%-90% for other languages, with the transcription process typically taking around 2.5 to 4 seconds. The multilingual translation module also performed well, with accuracy ranging from 87% to 92%. However, translating idiomatic or culturally specific phrases remains a challenge.

When it comes to animation mapping, the system was able to convert translated text into Indian Sign Language (ISL) animations with 93% accuracy. Most users found the animations intuitive and useful. The text-to-speech (TTS) functionality, with 97% pronunciation accuracy, provided accessible audio feedback for visually impaired users. However, some users mentioned the voice became monotonous during longer text passages, which could be improved for more engaging auditory feedback.

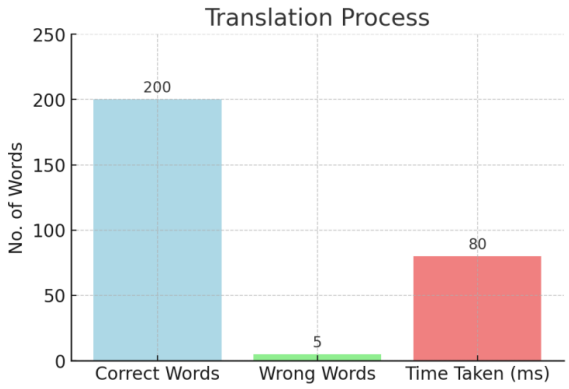

Fig 2. Translation Test

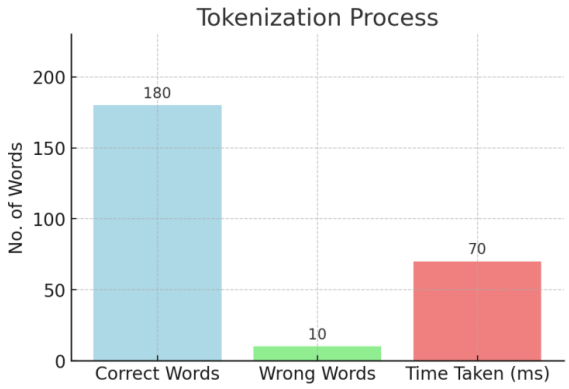

Fig 3. Tokenization Test

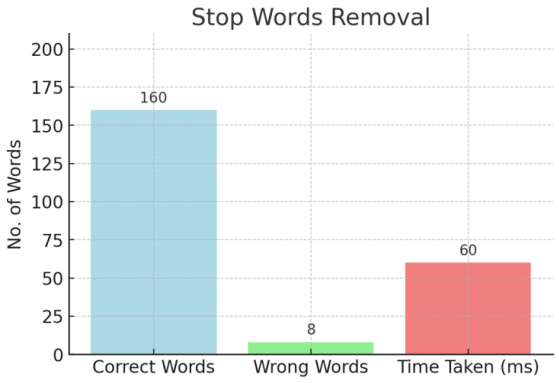

Fig 4. Stop Words Removal Test

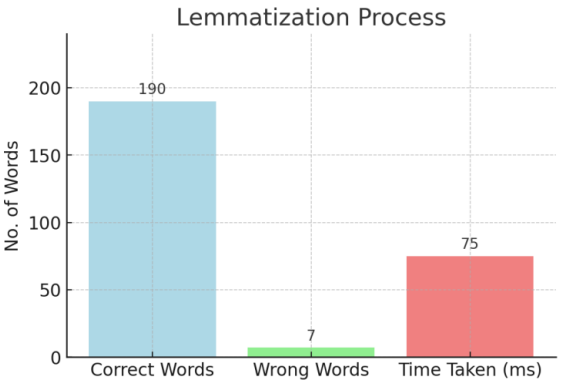

Fig 5. Lemmatization Test

The personalized dashboard kept users engaged, with 80% actively interacting with it for playback of transcriptions and animations. It was easy to navigate and had a user-friendly interface, though some users suggested adding more customization options. The system's ability to handle up to 100 concurrent users without a hitch shows good scalability. Despite this, when running resource-heavy tasks like translation and animation mapping at the same time, there were minor delays.

Some challenges persist, like handling complex sentence structures, accents, and real-time translation of certain expressions, but these are areas that can definitely be enhanced with further refinements. Going forward, integrating advanced models for better translation accuracy, expanding the animation library to cover more vocabulary, and enhancing the TTS feature with dynamic voices will improve the overall user experience.

CONCLUSION

Talking Fingers is a significant step toward bridging the communication gap between the deaf and hearing communities by leveraging modern technologies to provide real-time translation of spoken and typed text into Indian Sign Language (ISL) animations. The application effectively combines speech-to-text conversion, multilingual translation, and ISL animation mapping to address the challenges faced by the deaf community, including limited access to education, employment, and social inclusion. By supporting multiple Indian languages and providing a user-friendly web interface, the system caters to India's diverse linguistic landscape and fosters inclusivity.

The project demonstrates the potential of integrating the Web Speech API, googletrans, and NLP techniques to create a scalable and efficient solution for ISL translation. The results show high accuracy and usability, making Talking Fingers a reliable tool for real-world applications. Additionally, the system's modular architecture ensures that it can be expanded in the future to include more languages, a larger ISL vocabulary, and advanced features such as dynamic video synthesis and machine learning-based enhancements.

In conclusion, Talking Fingers provides an innovative and practical approach to promoting accessibility and empowering the deaf community. By bridging the communication gap, this project contributes to a more inclusive society, enabling greater interaction, understanding, and opportunities for the deaf population. Future work will focus on improving translation accuracy, expanding the ISL database, and exploring the use of advanced technologies to further enhance the system's capabilities.

REFERENCES

[1] Intwala, Nishi, Arkav Banerjee, and Nikhil Gala. "Indian sign language converter using convolutional neural networks." 2019 IEEE 5th international conference for convergence in technology (I2CT). IEEE, 2019

[2] Weisgrau, Maxine, Abraham Rosman, and Paula G. Rubel. The tapestry of culture: An introduction to cultural anthropology. Rowman & Littlefield, 2023.

[3] Kaviyadharshini, K., et al. "Conversion of Sign Language to Text and Audio Using Deep Learning Techniques." Advances in Information Communication Technology and Computing: Proceedings of AICTC 2022. Singapore: Springer Nature Singapore, 2023. 285-293

[4] Poonia, Ramesh Chandra. "LiST: A Lightweight Framework for Continuous Indian Sign Language Translation." Information 14.2 (2023): 79

[5] Neog, Anannya Priyadarshini, Arunabh Kalita, and Ms Nithyakani Pandiyarajan. "SPEECH/TEXT TO INDIAN SIGN LANGUAGE USING NATURAL LANGUAGE PROCESSING." (2023)

[6] Kumar, Chevella Anil, and Kancharla Anitha Sheela. "Development of a Speech to Indian Sign Language Translator." International Conference on Smart Computing and Communication. Singapore: Springer Nature Singapore, 2023

[7] Reddy, Bandi Rupendra, et al. "Indian Sign Language Generation from Live Audio or Text for Tamil." 2023 9th International Conference on Advanced Computing and Communication Systems (ICACCS). Vol. 1. IEEE, 2023

[8] Cherian, Dona Mary, and Jincy J. Fernandez. "Real-time sign language recognition and speech conversion using VGG16." International Journal of Computational Vision and Robotics 13.2 (2023): 174-185.

[9] Thorat, Sumitra Sambhaji. Sign Language Recognition and Text-toSpeech Translation. Diss. Dublin, National College of Ireland, 2023

[10] Joshi, Abhinav, Susmit Agrawal, and Ashutosh Modi. "ISLTranslate: Dataset for Translating Indian Sign Language." arXiv preprint arXiv:2307.05440 (2023).

[11] Nandi, Utpal, et al. "Indian sign language alphabet recognition system using CNN with diffGrad optimizer and stochastic pooling." Multimedia Tools and Applications 82.7 (2023): 9627-9648.

[12] Sonekar, S. V., et al. "Real-Time Sign Language Identification using KNN: A Machine Learning Approach." 2023 11th International Conference on Emerging Trends in Engineering & Technology-Signal and Information Processing (ICETET-SIP). IEEE, 2023.

[13] Rahaman, Muhammad Aminur, et al. "Computer vision-based six layered ConvNeural network to recognize sign language for both numeral and alphabet signs." Biomimetic Intelligence and Robotics (2023): 100141

[14] Yaseen, Zaher Mundher. "An insight into machine learning models era in simulating soil, water bodies and adsorption heavy metals: Review, challenges and solutions." Chemosphere 277 (2021): 130126.

[15] S.J. Basha, D. Veeraiah, G. Pavani, S.T. Afreen, P. Rajesh and M.S. Sasank, " A Novel Approach for Optical Character Recognition of Handwritten Telugu Alphabets using Convolutional Neural Networks", Second International Conference on Electronics and Sustainable Communication Systems(ICESC), Coimbatore, India,2021, pp.1494-1500.

[16] B Swathi, K Shalini, KN Prasanthi, " A Review on Steganography using Images", Asian Journal of Computer Science and Information Technology, Vol.2, Issue 8,2012.

[17] K. Kundeti, N.P., Naga Sai Damodhar, S., Gayathri, K., Bala Vamsi, M., Rahul Pramod, "Image caption to voice generation using deep learning", Journal of Advanced research in dynamical & control systems, vol.12(2),pp.897-904,2020.

[18] Hindman, Matthew. "Building better models: Prediction, replication, and machine learning in the social sciences." The Annals of the American Academy of Political and Social Science 659.1 (2015): 48- 62

[19] Hafeez, Rabab, et al. "Contextual Urdu Lemmatization Using Recurrent Neural Network Models." Mathematics 11.2 (2023): 435.