2024 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE) | 979-8-3503-0641-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/IITCEE59897.2024.10467692

Fostering Inclusive Communication: A Tool Integrating Machine Translation, NLP, and Audio-to-Sign-Language Conversion for the Deaf

Madhura Sen
School of Computer Science and Engineering
Vellore Institute of Technology
Vellore, India
ORCID: 0009-0006-1806-5592

R Rajkumar
School of Computer Science and Engineering
Vellore Institute of Technology
Vellore, India
vitrajkumar@gmail.com

***Abstract*— Approximately 70 million individuals worldwide grapple with deafness or muteness, presenting challenges in communication. This article presents a novel solution: an audio-to-sign-language converter. Sign language, a visual medium for speech, is pivotal in bridging this communication gap. This Python-based converter interprets user input, transforming it into gestures, facilitating interaction between the hearing and the deaf. It also serves as a valuable tool for learning sign language. Live voice input, captured through a microphone, undergoes advanced Natural Language Processing techniques to be converted into text. Recognized phrases trigger relevant gesture videos or display corresponding letters. The application also features a termination command, "goodbye." With potential applications in web and mobile platforms, two-way interaction, and video integration, this converter holds promise in enhancing accessibility for the deaf community, fostering inclusivity and effective communication. This paper is a significant step towards facilitating meaningful connections between individuals with disabilities and their non-disabled peers.**

Keywords*— Sign language, ISL vocabulary, accessibility, gestures, natural language processing, GIF, facial expressions, speech recognition

I. Introduction

Nearly 70 million individuals today grapple with deafness and muteness either from birth or due to unforeseen accidents. This unique challenge often leads to difficulties in interacting with the surrounding environment and communicating with those who do not share their condition. Bridging this gap is crucial for fostering meaningful connections between individuals with disabilities and their non-disabled counterparts.

Sign language offers a visual medium to convey speech through a vast vocabulary of gestures and signals. Before the advent of sign language, those with disabilities relied on facial expressions and rudimentary hand movements, making it arduous to articulate their thoughts and desires.

Sign language represents the most accessible and effective tool for communication in this context. Both disabled and non-disabled individuals must acquaint themselves with this visual lexicon. Unlike conventional speech, sign language eschews the need for oral communication, relying instead on a nuanced interplay of hand gestures and their orientations to convey messages. Each letter is assigned a distinct set of gestures, providing a structured framework for comprehension.

Sign language is a mother tongue for the deaf community, and the communication gap between deaf and hearing people must be bridged. Facilitating sign language has been simplified through the integration of gestures into a dedicated application. This application interprets user input and renders it in gesture form, acting as a bridge between the hearing and the deaf. Moreover, it serves as a valuable tool for learning sign language, offering two modes of input: live voice or data sourced from an embedded dataset.

Live voice input is captured through a microphone using PyAudio, subsequently recognized through Sphinx, and converted into textual form via advanced Natural Language Processing techniques. If the recognized text matches predefined phrases, relevant gesture videos are played; otherwise, the application displays corresponding letters in pictorial form. If the text detected from the speech says “goodbye”, the application exits. To enhance the interactivity of this application, we've incorporated training datasets that include both images and compact GIFs.

II. Literature Survey

Mariappan and Gomathi (2019) developed a real-time Indian Sign Language (ISL) recognition method using vision-based techniques, which involves data pre-processing (including frame resizing), morphological transformations, noise removal, contour detection, feature extraction, and fuzzy clustering. It employs the OpenCV library for video processing and classification. Achieving a 75% accuracy in gesture labeling, it outperforms comparable systems, recognizing 40 words of ISL in real time. Future enhancements include integrating Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) and expanding the vocabulary of recognized words [1].

Another system uses 3D animation to enable seamless communication between deaf and hearing individuals. It consists of two modules: speech-to-sign, which utilizes a speech-to-text library to map keywords to corresponding sign animations displayed by a 3D model, and sign-to-speech, which employs Microsoft Kinect V2's Continuous Gesture Builder for gesture recognition, constructing sentences from recognized gestures and converting them into speech. The system achieves notable accuracy rates of 84% for sign-to-speech and 87% for speech-to-sign modules. Sensitivity to non-US English accents highlights an area for potential improvement. [2].

Another model for hand gesture recognition to convert Bengali Sign Language to text uses CNNs and Support Vector Machines (SVMs). The process involves capturing and processing RGB images using a Kinect device, followed by joint identification and convex algorithm-based shape drawing. A five-layer neural network model is employed, with the final layer displaying the recognized gesture. Experimental results demonstrate an average accuracy of 84.11% for tested letters [3].

A system that translates double-handed ISL into text and speech uses the Minimum Eigen Value algorithm to detect feature points, focusing on corners for tracking. It analyzes image patches for corner detection. The Shi-Tomasi detector is employed for its scoring function, which marks pixels as corners based on intensity values within a small window. The system is trained with image features enabling efficient storage and computation. During real-time acquisition, a Logitech Web camera captures images, which undergo preprocessing to reduce noise and artifacts. Bare hands helped overcome the limitations of using data gloves. The extracted Min Eigen Value features are then matched with the database for recognition. The recognized text is further converted to speech using Text-to-Speech Synthesis [4].

Dutta and Bellary explored ISL, especially its unique grammar structure. Using MATLAB, ISL gestures are classified with machine learning techniques, specifically K Nearest Neighbors (K-NN) with K=1 and Back Propagation algorithms. Principal Component Analysis (PCA) is employed for dimensionality reduction. The system exhibits high accuracy, achieving 95.84% recognition for double-handed gestures and 94.88% for single-handed gestures. The conjugate gradient method is a commonly employed iterative algorithm for handling large sparse structures that may be impractical for direct implementation or other similar techniques such as the Chelsey decomposition [5].

Grif and Manueva developed computer animation for Russian Sign Language to enhance accessibility for individuals with hearing disabilities. A comparative analysis of Zardoz, TEAM, and ViSiCAST systems highlights the importance of integrating a semantic component for better performance. The study introduces a semantic analysis software unit to tackle lexical ambiguity, paving the way for more accurate translations. Additionally, a translation module is proposed to convert Russian text into Russian Sign Language, emphasizing syntactic and semantic transformations [6].

Gupta, Shukla, and Mittal focus on ISL, which includes both single-handed and double-handed static gestures, as well as a few dynamic ones. The proposed approach involves categorizing gestures, extracting features using HOG and SIFT descriptors, and applying a K-Nearest Correlated Neighbor algorithm for classification. The results show high accuracy in gesture recognition, especially with categorization. The technique is currently limited to detecting static movements. Potential enhancements include expansion to dynamic gestures and other non-verbal communication forms like mimics and expressions, and adapting the model for RGBD images from Kinect Sensors [7].

An approach by Patel and Ambekar (2017) focuses on image-based sign recognition using MATLAB, which involves capturing gestures with a webcam, preprocessing, and feature extraction using Hu moments. Classification is done using two methods: K-nearest neighbors (KNN) and Probabilistic Neural Network (PNN). PNN outperforms KNN in various evaluation parameters. Approximately 82% accuracy is achieved. The system converts sign gestures into both Hindi and English text and speech. Speech quality depends on recorded sound files. Future work includes extending it to other Indian languages, exploring applications for video conferencing, and adapting it for smartphones [8].

Santa, Tazreen, and Chowdhury (2017) present a framework for recognizing Bangladeshi Hand Sign Language (BdSL) from video input. It begins with converting frames to YCbCr color space and applies skin color segmentation. Hand regions are distinguished from other skin areas through labeling and filtering. Local Binary Pattern (LBP) is used for feature extraction, followed by classification using the Support Vector Machine (SVM). The framework demonstrates high accuracy in recognizing specific words and sentences. While the system excels in single and connected hand signs, it may face challenges with multiple unconnected hands. Addressing this limitation and enhancing low-contrast video quality are areas for future improvement [9].

Sengupta, Mallick, and Das (2019) aim to bridge the communication gap between deaf and hearing people using smart gloves, which can recognize finger-spelled words. The hardware components include a microcontroller, flex sensor, accelerometer, and Bluetooth module. The software development involves Arduino IDE for programming and MIT App Inventor for the graphical user interface (GUI) application on the smartphone. The system shows promising accuracy levels in recognizing finger-spelled gestures [10].

Suharjito, Gunawan, Thiracitta, and Nugroho used the i3d inception model, a modified version of inception v1, for 3D CNN architecture with the LSA64 dataset (10 vocabularies, 10 signers, 500 videos) to improve sign language detection. The complexity and diversity of Sign Language, varying by country and containing similar signs with different meanings, pose a challenge for Deep Learning models. This model faced challenges with overfitting on the datasets chosen, warranting further exploration for improved Sign Language recognition [11].

Suharjito, Thiracitta, Gunawan, and Witjaksono explore the application of Gaussian Hidden Markov Models (HMMs) for classifying sign language, specifically focusing on Argentina's Sign Language. A comprehensive analysis comparing preprocessing techniques, reveals that edge detection yields significantly higher accuracy than skin detection. In pre-processing the skin discovery, the Multinomial Hidden Markov Model was more accurate than the Gaussian Hidden Markov Model. The study acknowledges limitations related to dataset quality, particularly the use of gloves by signers, impacting feature accuracy. Future work includes enhanced noise reduction, dataset collection, and feature extraction techniques [12].

Truong, Yang, and Tran (2016) developed a system capable of translating sign language into text and speech, focusing on American Sign Language (ASL). The system employs a robust Haar-like features-based AdaBoost classifier for hand detection in live videos, a challenging task due to the dynamic nature of sign language interpretation. It aims to recognize hand signs correctly, convert these recognized signs into letters, combine them into words and sentences in textual format, and then translate them to audio format. An impressive recognition rate of 98.7% for 26 ASL hand postures was achieved [13].

Warrier, Sahu, Halder, Koradiya, and Raj (2016) present a novel approach utilizing LabVIEW, a graphical programming platform, in conjunction with smartphone cameras for real-time sign language conversion. The system captures hand gestures through the DroidCam Android app, processes the acquired images for enhanced recognition, and employs geometric matching for gesture recognition. The recognized gestures are then converted into textual and audio formats, enabling effective communication between deaf and hearing individuals [14].

Yang, Sclaroff, and Lee address the challenge of detecting and recognizing signs within a continuous gesture stream. Their method employs a conditional random field (CRF) model with adaptive thresholding to differentiate between signs in a specified vocabulary and nonsign patterns. It also incorporates a short-sign detector, hand appearance-based sign verification, and subsign reasoning techniques to further enhance accuracy. Experimental results indicate an 87.0 percent spotting rate and a 93.5 percent recognition rate for continuous and isolated data, respectively, surpassing the performance of conventional CRFs [15].

Zheng, Chen, Wu, Shi, and Kamal investigated the often-overlooked role of facial expressions in enhancing sign language translation accuracy. Their proposed framework, comprising a Semantic Focus of Interest Network with a Face Highlight Module (SFoI-Net-FHM) and a Machine Translation Network (MT-Net), deliberately integrates facial expressions to improve translation precision. Utilizing a Multi-stream Architecture for both facial and global features and a Region of Interest (RoI)-based Multi-region Architecture employing object detection for facial and body regions, the SFoI-Net-FHM demonstrated significant improvements on the RWTH-PHOENIX-Weather-2014T dataset, surpassing baseline performance [16].

Kan et al. propose the Hierarchical Spatio-temporal Graph Neural Network (HST-GNN) as an innovative solution to overcome limitations in existing sign language translation methods. It introduces a sophisticated architecture employing hierarchical spatio-temporal graphs. Unlike conventional models, this method utilizes graph convolution and self-attention mechanisms to represent signing poses effectively. The encoder-decoder framework incorporates high-level and fine-level graphs, enabling the capture of nuanced body region interactions. Evaluations on benchmark datasets demonstrate that the HST-GNN achieves state-of-the-art performance in sign language recognition and translation. Ablation studies further dissect the contributions of spatial, temporal, and hierarchical modules, highlighting the significance of each in enhancing overall performance [17].

Irasiak, Kozak, Piasecki, and Steclik conducted a comprehensive analysis of action units (AUs) to capture facial expressions focusing on Polish Sign Language. The study carefully examined the entropy of AUs utilizing OpenFace 2.0 for AU detection and emphasizing facial expressions in avatar control during signing. The results demonstrated that averaging over five frames significantly improved the precision of facial expression analysis. Correlation analyses uncovered classical and additional relationships between AUs, emphasizing the crucial role of facial expressions in sign language recordings. The research suggests the potential utility of AUs in linguistic research, corpus studies, and sign language dictionary development while paving the way for automated sign language annotation and translation tools. However, the study acknowledges the need for further exploration, particularly in addressing the intricacies of mouth actions in sign language expressions [18].

III. Architectural Design

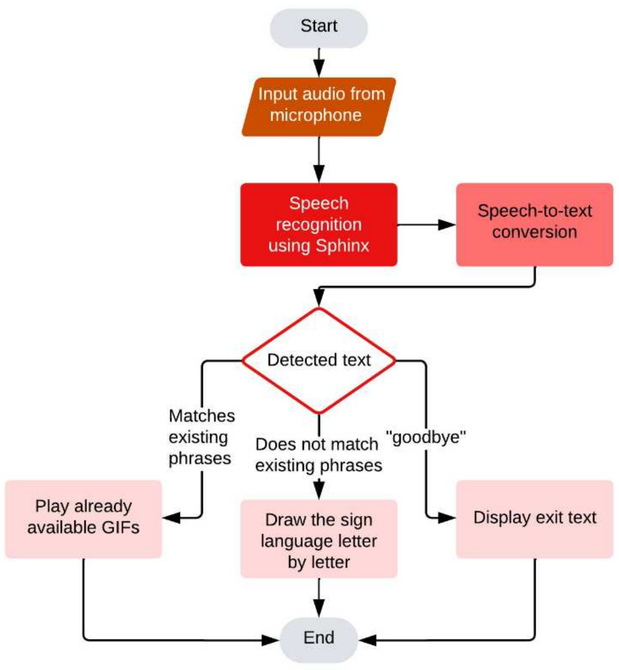

Start
Input audio from microphone
Speech recognition using Sphinx
Speech-to-text conversion
Detected text
Matches existing phrases
Does not match existing phrases
"goodbye"
Play already available GIFs
Draw the sign language letter by letter
Display exit text
End

Fig. 1. Architectural Design

IV. Implementation

This application can be run either through the command prompt or using Anaconda. Start by navigating to the directory where the project files are stored and run the main.py Python file. This will initiate the application and display the message “Say something.”

In our implementation, we load all the necessary libraries, including speech recognition, NumPy, matplotlib.pyplot, cv2, and others. Once the application is launched, three options are presented: live recording, recorded voice, and the exit option, “All done.” For the recorded voice option, users can provide pre-existing voice clips, while live recording captures user input via the microphone using the built-in recognizer function, a Python standard library.

Recorded voice clips are stored in temporary memory, and we apply text preprocessing using natural language processing techniques. If the application encounters difficulty recording the voice, it displays a user-defined message, typically, “Could not listen.” We then perform dictionary-based machine translation on the detected text. If a word is not found in our trained dictionary, we spell it out using sign language, which we have already incorporated for the entire alphabet (A to Z).

To enhance user interaction and ease of use, we integrated GIFs and short video clips depicting common phrases like “Hello,” “Good Morning,” and “What is your name.” This streamlines the process for frequently asked questions and compliments. This serves as the primary objective of our application.

For proper termination of the application, we implemented a specific command, “Goodbye.” The application exits when this command is received during the audio input phase.

A. *Algorithm*

```
1. Application starts.
2. Capture user voice input via microphone.
        1.1 Listen for a specific duration.
        1.2 Convert captured voice to text using
            speech recognition libraries.
3. Save the voice in the temporary memory.
4. Convert the entire text string to lowercase for
   further processing.
        4.1 To proceed with further manipulation,
            convert the entire string of text to
            lowercase letters.
5. Search each character of the input text in the data
   set.
        5.1 If the text is “goodbye”, exit the
            program.
        5.2 If not, search the word in the predefined
            dictionary images and GIFs.
        5.3 If not found, spell the word using
            symbols with a delay in image display
            actions.
6   Repeat from step 3 till the speech ends.
7   If an error occurs in Step 2, display: "Could not
    listen."
```

B. *Modes of usage*

- Live Voice: Enables user interaction to receive Indian Sign Language visuals from the provided speech.
- All Done: Allows the user to exit the application

V. Results

Fig. 2 shows the homepage of the Hearing Impairment Assistant. It converts live audio to Indian Sign Language. It provides us with two options. The first option, ‘Live voice,’ takes the input audio from the end user and converts it into its respective GIF or Image. The other option, ‘All Done,’ is used to exit from the interface once the session is completed.

Once the user has clicked the Live Voice button, the interface starts recognizing the voice input. As the system begins recognizing, it prints “I am Listening” on-screen. Once the phrase has been identified, the interface prints “You Said: the word recognized.” In fig. 3, the input phrase is “good morning”.

On recognizing the phrase, the interface goes through the dataset to search for the input phrase. Once the respective text has been matched with the GIF in the dataset, it automatically pops up on the screen, as shown in fig. 4.

If the respective GIF is not present in the dataset for the input phrase, the software starts recognizing the phrase letter-by-letter and then prints the sign images of the individual letters iteratively. In fig. 5, the phrase has been recognized as “week”, and the respective images are popped up iteratively.

To exit from the interface, the user has to vocalize “goodbye.” The interface prints the end statement “Thank you for using the Hearing Impairment Assistant. Have a Great Day!!” The end user will then be taken to the initial homepage in which the user can click “All Done!” to close the software as in fig. 6.

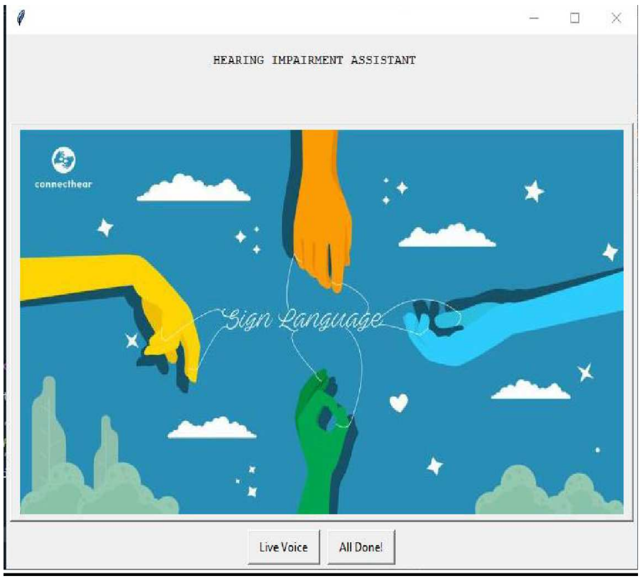
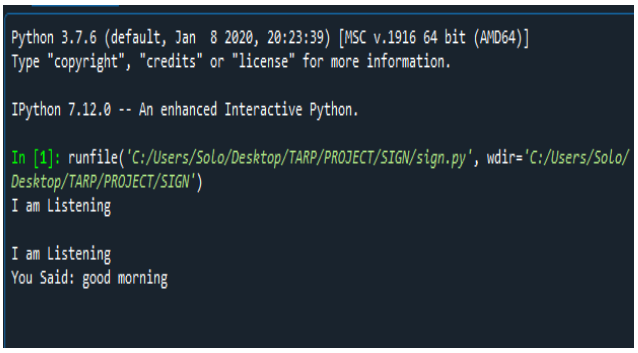


Fig. 2. Homepage



```
Python 3.7.6 (default, Jan  8 2020, 20:23:39) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.12.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/Solo/Desktop/TARP/PROJECT/SIGN/sign.py', wdir='C:/Users/Solo/
Desktop/TARP/PROJECT/SIGN')
I am Listening

I am Listening
You Said: good morning
```

Fig. 3. Live voice recognition

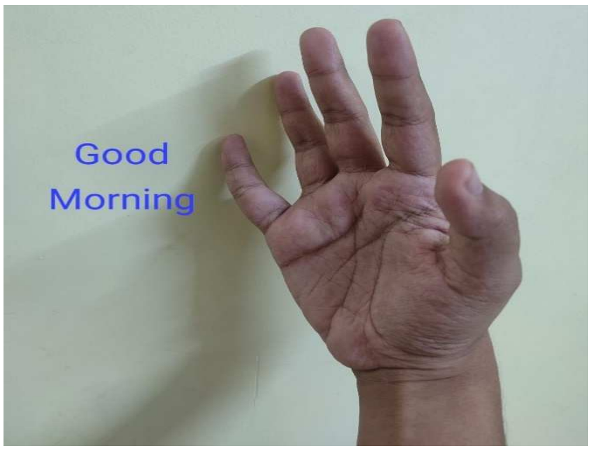
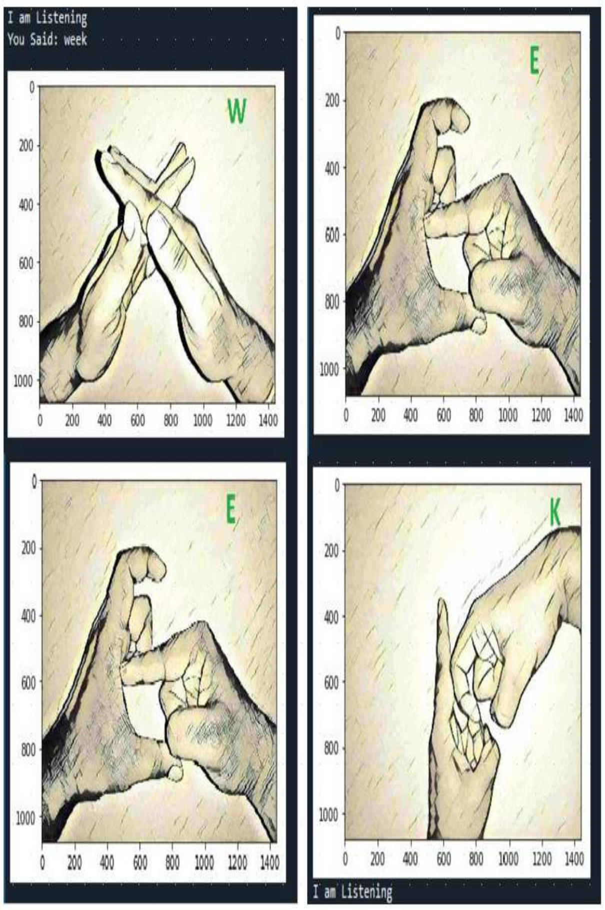
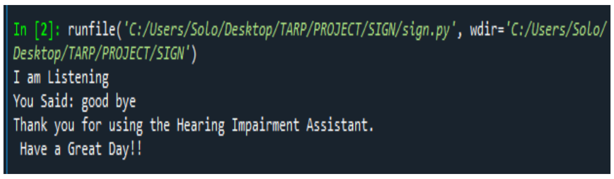


Fig. 4. GIF popup of recognized text


I am Listening
You Said: week

I am Listening

Fig. 5. Iterative image popup of phrase



```
In [2]: runfile('C:/Users/Solo/Desktop/TARP/PROJECT/SIGN/sign.py', wdir='C:/Users/Solo/
Desktop/TARP/PROJECT/SIGN')
I am Listening
You Said: good bye
Thank you for using the Hearing Impairment Assistant.
 Have a Great Day!!
```

Fig. 6. Exit process

VI. Conclusions and Future Work

Interacting with individuals with challenges in communication, like the deaf community, can be difficult due to the time-consuming process of learning sign language. An audio-to-sign-language converter addresses this issue by providing deaf individuals with a way to access information in Indian Sign Language (ISL) efficiently. This paper offers a sustainable solution for capturing the entire ISL vocabulary. Deaf individuals often face limitations in communication, relying on interpreters or visual communication methods. This system aims to alleviate the dependence on interpreters by converting speech into text and presenting the output as either text or motion.

Future scope includes expanding the application to a web-based and mobile app version, extending its accessibility to a broader audience and thus providing a practical way for disabled individuals to communicate. Furthermore, enhancing the application could involve adding two-way interaction features, such as video-to-sign language conversion and sign-language videos producing audio/text outputs. Multilingual support is also a useful enhancement, enabling compatibility with many different sign languages. Additionally, incorporating video aspects into communication can further improve the effectiveness of the application. The application can also be integrated with educational and online learning platforms, providing a valuable resource for sign language education.

References

[1] H. Muthu Mariappan and V. Gomathi, "Real-Time Recognition of Indian Sign Language," 2019 International Conference on Computational Intelligence in Data Science (ICCIDS), Chennai, India, 2019, pp. 1-6, doi: 10.1109/ICCIDS.2019.8862125.

[2] M. Ahmed, M. Idrees, Z. ul Abideen, R. Mumtaz and S. Khalique, "Deaf talk using 3D animated sign language: A sign language interpreter using Microsoft's kinect v2," 2016 SAI Computing Conference (SAI), London, 2016, pp. 330-335, doi: 10.1109/SAI.2016.7556002.

[3] A. R. Chowdhury, A. Biswas, S. M. F. Hasan, T. M. Rahman and J. Uddin, "Bengali Sign language to text conversion using artificial neural network and support vector machine," 2017 3rd International Conference on Electrical Information and Communication Technology (EICT), Khulna, 2017, pp. 1- 4, doi: 10.1109/EICT.2017.8275248.

[4] K. K. Dutta, S. K. Raju K., A. Kumar G.S. and S. A. Swamy B., "Double handed Indian Sign Language to speech and text," 2015 Third International Conference on Image Information Processing (ICIIP), Waknaghat, 2015, pp. 374-377, doi: 10.1109/ICIIP.2015.7414799.

[5] K. K. Dutta and S. A. S. Bellary, "Machine Learning Techniques for Indian Sign Language Recognition," 2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC), Mysore, 2017, pp. 333-336, doi: 10.1109/CTCEEC.2017.8454988.

[6] M. Grif and Y. Manueva, "Semantic analyses of text to translate to Russian sign language," 2016 11th International Forum on Strategic Technology (IFOST), Novosibirsk, 2016, pp. 286-289, doi: 10.1109/IFOST.2016.7884107.

[7] B. Gupta, P. Shukla and A. Mittal, "K-nearest correlated neighbor classification for Indian sign language gesture recognition using feature fusion," 2016 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, 2016, pp. 1-5, doi: 10.1109/ICCCI.2016.7479951.

[8] U. Patel and A. G. Ambekar, "Moment Based Sign Language Recognition for Indian Languages," 2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA), Pune, 2017, pp. 1-6, doi: 10.1109/ICCUBEA.2017.8463901.

[9] U. Santa, F. Tazreen and S. A. Chowdhury, "Bangladeshi hand sign language recognition from video," 2017 20th International Conference of Computer and Information Technology (ICCIT), Dhaka, 2017, pp. 1-4, doi: 10.1109/ICCITECHN.2017.8281818.

[10] A. Sengupta, T. Mallick and A. Das, "A Cost Effective Design and Implementation of Arduino Based Sign Language Interpreter," 2019 Devices for Integrated Circuit (DevIC), Kalyani, India, 2019, pp. 12-15, doi: 10.1109/DEVIC.2019.8783574.

[11] Suharjito, H. Gunawan, N. Thiracitta and A. Nugroho, "Sign Language Recognition Using Modified Convolutional Neural Network Model," 2018 Indonesian Association for Pattern Recognition International Conference (INAPR), Jakarta, Indonesia, 2018, pp. 1-5, doi: 10.1109/INAPR.2018.8627014

[12] Suharjito, N. Thiracitta, H. Gunawan and G. Witjaksono, "The Comparison of Some Hidden Markov Models for Sign Language Recognition," 2018 Indonesian Association for Pattern Recognition International Conference (INAPR), Jakarta, Indonesia, 2018, pp. 6-10, doi: 10.1109/INAPR.2018.8627031.

[13] V. N. T. Truong, C. Yang and Q. Tran, "A translator for American sign language to text and speech," 2016 IEEE 5th Global Conference on Consumer Electronics, Kyoto, 2016, pp. 1-2, doi: 10.1109/GCCE.2016.7800427.

[14] K. S. Warrier, J. K. Sahu, H. Halder, R. Koradiya and V. K. Raj, "Software based sign language converter," 2016 International Conference on Communication and Signal Processing (ICCSP),

Melmaruvathur, 2016, pp. 1777-1780, doi: 10.1109/ICCSP.2016.7754472

[15] H. Yang, S. Sclaroff and S. Lee, "Sign Language Spotting with a Threshold Model Based on Conditional Random Fields," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 7, pp. 1264-1277, July 2009, doi: 10.1109/TPAMI.2008.172.

[16] J. Zheng, Y. Chen, C. Wu, X. Shi, and S. M. Kamal, "Enhancing neural sign language translation by highlighting the facial expression information," *Neurocomputing*, vol. 464, pp. 462–472, Aug. 2021. doi:10.1016/j.neucom.2021.08.079 .

[17] J. Kan, et al., "Sign Language Translation with Hierarchical Spatio-Temporal Graph Neural Network," in 2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 2022 pp. 2131-2140, doi: 10.1109/WACV51458.2022.00219

[18] A. Irasiak, J. Kozak, A. Piasecki, and T. Stęclik, "Processing real-life recordings of facial expressions of Polish sign language using action units," Entropy, vol. 25, no. 1, p. 120, Jan. 2023. doi:10.3390/e25010120