Grenze International Journal of Engineering and Technology, January Issue

TouchSign Bridge: Overall Solution for Visual and Speech-Impaired People

Durvesh Chaudhari[1], Adrika Dikshit[2], Jay Kolte[3] and Deepak Karia[4]
[1-4] Sardar Patel Institute of Technology/Electronics Engineering, Mumbai, India
Email: durvesh.chaudhari@spit.ac.in, adrika.dikshit@spit.ac.in, jay.kolte@spit.ac.in, deepak_karia@spit.ac.in

***Abstract*— TouchSign Bridge is an innovative solution with the dual functionality of translating Indian sign language to text and audio as well as converting the text to braille. The software aids inclusivity for individuals who face extraordinary challenges while establishing a social connection through verbal communication. It is a web-based application that seamlessly facilitates communication between the specially-abled particularly those with visual and auditory impairments with the general population The system makes use of the Convolutional Neural Network (CNN), a deep learning algorithm commonly used in image processing and recognition tasks. The CNN model is trained on a large dataset of sign language gestures captured through videos or images, which includes a variety of gestures, variations, and lighting conditions and shows a testing accuracy of 80%. Google's Text-to-Speech API is being used for audio implementation. Python dictionaries are used for textual to-braille conversion. React was employed on the front end and Flask on the server end to integrate this functionality into a web application.**

***Index Terms*—Convolutional Neural Networks, Sign language, Real-time translation, Braille, Classification, Deep Learning, Python.**

I. INTRODUCTION

Communication stands as the cornerstone of human interaction, a pivotal driver in the evolution of societies. However, within our communities exist individuals whose world revolves around languages comprehensible to only a selected few. Among these are Braille, understood solely by the visually impaired, and sign language, grasped exclusively by the deaf and mute. Despite the significance of these communication forms, their usage is limited, with a staggering 63 million deaf and hard-of-hearing individuals in India alone [1].

For individuals who are deaf or mute, sign language is an indispensable means of communication, in addition to technologies like hearing aids and lip-reading techniques. Nonetheless, the general public's lack of knowledge of sign language and braille hinders effective communication with deaf and blind people. Moreover, the various sign languages that are employed differ depending on the region thus lacking a globally accepted form.

In parallel, the visually impaired face barriers in accessing readily available information to the general public. However, the omnipresence of technology in modern civilization offers a glimmer of hope. By harnessing technology, communication gaps are bridged. Avenues are provided for the differently abled to lead more independent and inclusive lives. This synergy between the communities enhances communication, education and inclusion of both groups and fosters a more accessible and compassionate society.

Grenze ID: 01.GIJET.11.1.45

TouchSign Bridge is an advanced technology that facilitates smooth discourse by serving as a real-time translator for the deaf and mute. Additionally, it enables the conversion of text inputs into Braille, giving blind and visually impaired individuals access to a plethora of previously unattainable information. The incorporation of web-based applications that are easy to use ensures that this technology is accessible.

This kind of technology utilization not only enhances the lives of people with disabilities tremendously but also helps businesses. By using better communication techniques, organizations can effectively market their products and services to a larger audience, which also includes individuals with disabilities. In addition to improving people's lives, this inclusive strategy helps firms reach a wider and more varied customer base.

The rest of the document is structured as follows: Related work is addressed in Section 2. The system's technologies are covered in Section 3. The methodology for system implementation is fully explained in Section 4. The components of implementation and working are highlighted in Section 5. The predictions made by the model are the topic of Section 6. Results and a discussion of the developed system are the main topics in Section 7. The study's conclusions are included in Section 8.

II. Related work

In the past few years, there have been many advancements in sign language recognition systems. The referred papers address many challenges [2] Rumama et al. [3] created a sign language recognition system where data acquisition is done using glove-mounted sensors and subsequent classification using Neural Networks, Support Vector Machines, and Hidden Markov Models. In this, the feature extraction is carried out by capturing webcam frames, applying Gaussian blur, and implementing thresholding. Similarly, Hasan et al. [4] created a model for Bangla sign language recognition using the Support Vector Machine (SVM) classifier, achieving a recognition rate of 86.53%, along with audio output to help users understand the interpreted signs.

Many papers have focused on the integration of machine learning techniques, especially Convolutional Neural Networks (CNNs) for enhancing sign language recognition accuracy [5]. Chaturvedi et al [6] developed a real-time communication system with the help of deep learning and computer vision, with an accuracy of 99% with their CNN model. Thakur et al [7] gave a two-way system between sign language and text using a CNN-based model. Then in another paper, Singh et al [8] created a convolutional neural network for dynamic sign recognition, where some of the challenges are the varying video quality and the bad lighting conditions. Potentially these papers suggest that the deep learning techniques can overcome the challenges posed by variations in sign language gestures.

Some people also suggested a few hardware integrations and advancements with the sign language recognition systems [9]. Rajamohan et al. [10] presented a glove-based translation system that uses flex sensors and accelerometers for precise detection of the gestures. This approach helped convert the sign language to text and also incorporate the text into speech, providing a proper solution for the deaf and the mute. In another related work, Farhan et al. [11] developed an American sign language detection system using the MediaPipe framework and Long Short-term Memory (LSTM), with an accuracy of 97.2%. Such hardware-centric solutions make these models more accessible and user friendly [12, 13].

The study also focuses on the braille text-to-speech [14], where the Braille is converted to English to eliminate the need for a separate braille learning process. Users are allowed multiple languages, where they can upload scanned braille documents. These braille letters are recognized and translated into English, thus converting to speech using the google-text-to-speech (GTTS).

III. Technologies used

A. OpenCV

OpenCV is a common open-source computer vision toolkit that provides an extensive range of image and video processing capabilities [15]. Particularly, OpenCV (cv2) is used in this project to do hand detection, cropping, scaling, and image display. The processes involved in capturing webcam photos include 'cv2.VideoCapture', 'cv2.flip', 'Hand Detector' for identifying hands from 'cvzone', 'cropping, resizing, and pasting into a white backdrop', and 'cv2.imshow' for displaying the result.

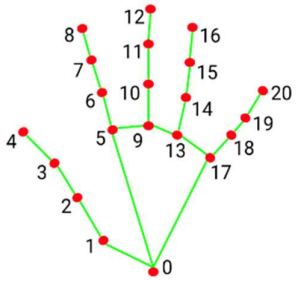
B. NumPy

The NumPy [16] library for Python is an acronym for "Numerical Python." It is used in this project to generate an array of zeros that serves as the cropped hand image's white background. After resizing, the background picture is placed into the white background array's center. Other numerical operations that NumPy can do with

the arrays include figuring out the dimensions and scaling factor for the cropped image, figuring out the offset for centering the cropped image on the white background, and filling the background array with white pixels. In conclusion, the provided code uses NumPy for numerical operations on arrays and image processing tasks.

C. HandTracking Module's HandDetector

Using Mediapipe's hand identification model, the HandDetector module analyzes the video frames to identify the regions of the frame that represent the hands [17]. As seen in Figure 1, the hand landmarks identify 21 coordinates on the observed region. It also provides an array of parameters and settings to optimize the hand-detection algorithm's performance in different situations. This code detects the presence and location of hands in a video stream captured by the computer's webcam using the HandDetector from CV Zone's HandTracking Module. After the hands are detected, the bounding box of each hand is located, and the image inside is scaled and cropped to a preset size [23].



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 1. MediaPipe Landmark [17]

D. Classification Module's Classifier

The Classifier in the CV Zone's Classification Module is a computer vision package that facilitates image processing and artificial intelligence operations. This project uses it to automatically classify hand motions in real time. A pre-trained deep learning model in the module is capable of identifying hand motions from an image. The classifier predicts the matching label after receiving an image of the hand gesture as input. The code uses the anticipated label to create a text message over time, that is shown on the output video.

E. OS

The 'OS' [18] module provides a portable way of using operating system-dependent functionality. The module is used in this project to create directories to store the images that are being captured by the program. Specifically, the code creates directories for each capital letter from A to Z and each number from 1 to 9 in the specified directory path. It is also used for accessing these paths.

F. Google's Teachable Machine

Google Teachable Machine uses transfer learning to create new models based on pre-trained neural networks [19]. Transfer learning speeds up the training of custom models on a smaller dataset by utilizing pre-trained model characteristics. Speech recognition, object identification, natural language processing, and image classification are just a few of the many model types it supports. For tasks like pose and emotion recognition, pre-built models are available, and trained models can be exported as TensorFlow Keras (.h5) files for application in software applications.

G. React

React is a free and open-source front-end JavaScript library for building user interfaces based on components [20]. It is used in the project to create client-side applications. The usage of react in our project has aided us in making reusable UI components which is highly advantageous when building various sections for sign language-to-speech translation, text-to-speech and text-to-braille conversions, with each functionality being encapsulated within its component.

H. Flask

Flask is used in the backend to handle requests from each application, process data, interact with databases and respond to necessary information [21]. The flask sends back JSON responses of the image data in case of sign language to speech and converts braille text in case of text to braille. It handles HTTP requests apart from

serving as an API endpoint, providing functionalities such as gesture classification and text-to-braille conversion by responding with data in a structured JSON format.

I. Google Text-to-Speech

Google Text to Speech (gTTS) is a service provided by Google that allows users to convert text to naturally sounding speech [22]. In the project, the alphabets, phrases or numbers generated are stored in a list and then joined and stored. This joined output is the text that gets converted into speech. The generated speech is stored in an MP3 file named 'text2speech.mp3'.

IV. METHODOLOGY

The web application can accept input in two forms: real-time video and text. The input is sent to the backend for processing, and the output is displayed. Figure 2 shows the flow of input from the frontend to the backend.

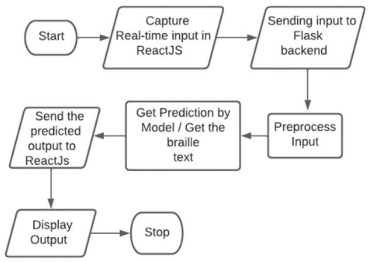


Figure 2. Flow Chart of System Implementation

The video input captures the video at a frame rate of 30fps (frames per second), and the image frames are sent to the backend for preprocessing and prediction. The deep learning algorithm chosen for classifying the output is CNN (Convolutional Neural Network).

The model is trained to recognize static signs that do not require dynamic movements to logically define the action. It identifies signs which include numbers from '0' to '9', alphabets from 'A' to 'Z' and greeting signs such as, 'How are you?', 'Morning', 'Afternoon', 'Hello', 'Good' etc. The cvzone's hand detector module was utilized, which is built on the mediapipe's framework. The dataset used has a total of around 28,700 images, which is divided into a ratio of approximately 70:30 as training is to testing [23].

After predicting the output for an image frame, text is generated and then converted to audio using the 'gTTS' Python library [22]. Additionally, the text is converted to braille using the following methodology.

The second kind of input taken is in text format and it is sent to the backend for conversion with the braille output being displayed.

V. IMPLEMENTATION

The web application is built using 'ReactJs', a frontend JavaScript library, and 'Flask' as the backend framework. Flask is chosen due to the Python implementation of the prediction algorithm. The input is taken in the form of text or real-time video. Figure 3 shows the frontend of the web application.

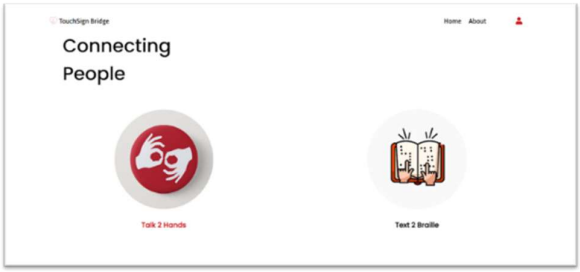


Figure 3. Webapp Frontend

For real-time video input, the captured image frames are sent to the backend for preprocessing. Preprocessing is performing various operations, like resizing, normalization and noise reduction, to enhance data quality and make it more suitable for effective model training. A bounding box [23] is created using cvzone's HandDetector module, to get an appropriate processed image. The resulting image contains a hand(s) image with a uniform dimension of 300 x 300 px as shown in Figure 4. For prediction, the resulting image frame is passed to the model.

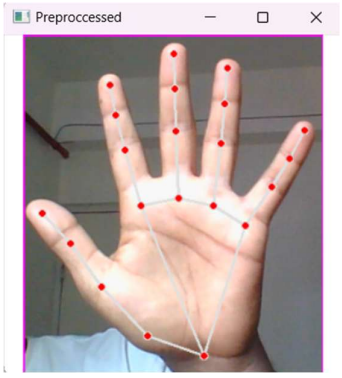


Figure 4. Processed image with white background

To create a sentence, a time frame of 5 seconds is considered. With a video capture rate of 30 fps, 30 frames are predicted every second. In the time frame of 5 seconds, 150 frames are predicted and stored in a Python data-structure 'list'. After every 5 seconds, the output that has been predicted the maximum number of times is assumed to be the correct output and is added to the sentence. The sentence formed is then converted to audio using the 'gTTS' Python library [22].

The textual input or the text formed in the above process is converted to Braille. A Python data-structure 'dictionary' is created where the keys include all the English alphabets (both lowercase and uppercase), numbers from 0 to 9, and various punctuation marks and symbols such as a comma, period, question mark, exclamation mark, apostrophe, double quotes, hyphen, underscore, parentheses, ampersand, semicolon, colon, forward slash, backslash, and the "at" symbol. The text is then converted character by character to its corresponding Braille value. The resulting Braille output is then displayed.

VI. PREDICTION AND RECOGNITION

The image frames are sent to the model for prediction. Figures 5 and 6 show the accurate prediction for the sign of the letter '5' and the phrase 'Good' respectively on the terminal.

Figure 5. Detection of the number '5' sign with the label

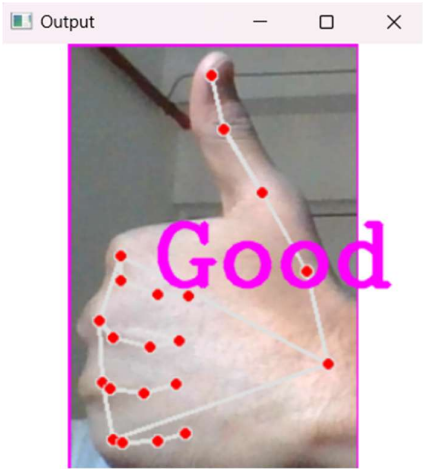


Figure 6. Detection of the word ‘Good’ with the label

Figures 7 and 8 show the accurate conversion of sign to text and text to braille respectively

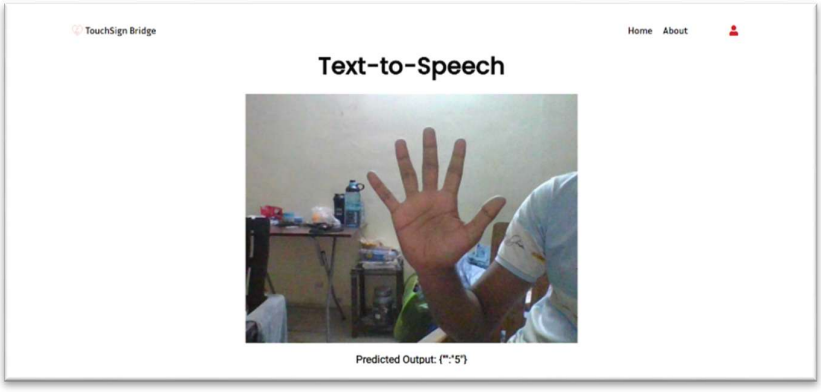


Figure 7. Detection of the number ‘5’ on the web-app

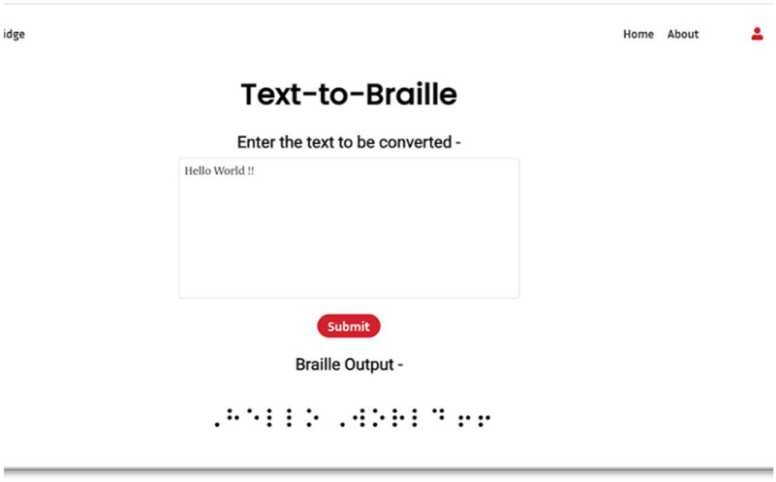


Figure 8. Text to Braille Result

VII. RESULTS AND DISCUSSION

Once a character is predicted, a sentence is formed and converted into audio and braille.
The model is tested with the 8,200 images of the testing dataset. Tables 1, 2, and 3 show the accuracy of numbers, phrases, signs, and alphabets. The macro-testing accuracy achieved by the model is 80.228%, whereas the macro-training accuracy achieved is 97.241%. The highest accuracy of 100% was achieved for signs 1, 9, B, G, H, O, Q, R, W, and Y. Whereas, the lowest accuracy was observed for sign 6, with an accuracy of 12.1%. Except for a few labels, the accuracy achieved per sign is observed to be above 60%. The credit for this accuracy goes to the deep learning classification algorithm used i.e., CNN.

TABLE I. ACCURACY FOR PREDICTION OF NUMBERS

Sign	**Accuracy (%)**
0	91.6
1	100
2	55.4
3	94.8
4	42.8
5	86.8
6	12.1
7	78.2
8	65.2
9	100

TABLE II. ACCURACY FOR PREDICTION OF PHRASES

Sign	**Accuracy (%)**
Good	58.4
Evening	98.1
Hello	34.9
Morning	99.1
How are you	93.1

TABLE III. ACCURACY FOR PREDICTION OF LETTERS

Sign	**Accuracy (%)**
A	97.7
B	100
C	94.5
D	83.4
E	83.5
F	94.8
G	100
H	100
I	54.1
J	97.1
K	78.8
L	12.5
M	93.4
N	39.5
O	100
P	91.3
Q	100
R	100
S	98.9
T	98.7
U	43.4
V	17.6
W	100
X	99.5
Y	100
Z	99.1

Along with the Accuracy, some other evaluation metrics are also considered. The Receiver Operating Characteristic (ROC) curve is a valuable tool to assess the performance of the model. It also helps to understand

the trade-offs between true positives (sensitivity) and false positive rates (1 - sensitivity) for each class. Figure 9 shows the ROC curve obtained. It shows the overall performance of 41 classes used. As different classes have different testing accuracies, the curve for each class differs. The classes having curves near the upper-left corner have an overall better performance.

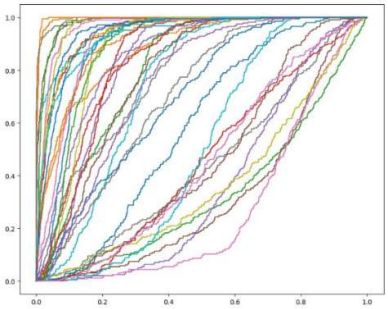


Figure 9. ROC Curve

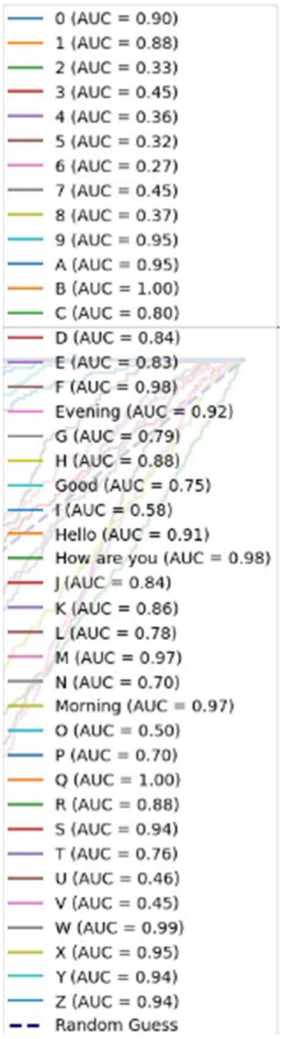


Figure 10. Labels for ROC Curve

The Precision value is defined as the accuracy of positive predictions made by a model. The precision value obtained for the model is 0.881 i.e., 88.1% of instances that were predicted as positive by the model are positive.
The Recall value measures the proportion of true positive instances that the model correctly identifies among all actual positive instances in the dataset. This value for the model is 0.797, or 79.7% of all the positive instances were correctly identified by the model.
The F1 score is the harmonic mean of the Precision and the Recall value. A higher F1 score indicates a good balance between the Precision and the Recall Value. The recall value obtained for the model is 0.801 or 80.1% indicating a good balance between the Precision and Recall value for the model.
Figure 11 shows the Precision, Recall and F1 score values.

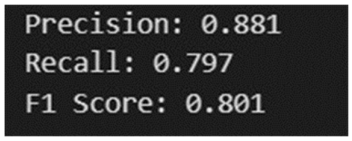


Figure 10. Values of Evaluation Metrics Used

VII. Conclusion and future scope

This study aims to develop a user-friendly web application, using React and Flask, that bridges the communication gap between deaf, mute, and blind individuals with those who are not, by converting Indian Sign Language (ISL) to text, audio and braille. To achieve this, a Convolutional Neural Network (CNN) model was trained on a dataset of approximately 21,000 diverse images, consisting of 41 categories of static signs commonly used in ISL, such as numbers, alphabets, and common phrases. The model was trained and tested on a dataset containing image frames with different backgrounds, lighting conditions, and hand shapes to avoid bias in the model. The background noise was minimized i.e., the image frames consist only of functional hands.
The transfer learning techniques were used, provided by Google's Teachable Machine to train the model. On testing the model, it achieved an overall macro-accuracy of 80.228% on a testing dataset of approximately 8,200 images. Whereas, it achieved an accuracy of 97.241% on the training dataset of roughly 20,500 images. The model's accuracy can be further improved by incorporating a larger and more diverse dataset for training.
In future, an actual CNN model will be deployed to achieve enhanced performance by deploying artificial neurons. This will help the model identify the gestures accurately. Additionally, the dynamic signs will also be included in the dataset using video data, as most of the communication in ISL is based on hand movements and not static signs.
The utilization of the backend for prediction repeatedly slows down the prediction process. The video frame capture rate is 30fps. Hence, passing 30 frames in a second and getting their prediction, requires the module to restart repeatedly which hampers the speed of prediction. Therefore, in future, a new method of implementation will to used which do not require the model to start each time it receives an image frame.
Ultimately, the main focus would be on improving the model's accuracy and building a ready-to-deploy product.

References

[1] Varshney, Saurabh (2016-04-01). "Deafness in India". Indian Journal of Otology. 22 (2): 73. doi:10.4103/0971-7749.182281. ISSN 0971-7749. S2CID 78805217.
[2] R, S. et al. (2022) 'Indian sign language to speech conversion using Convolutional Neu-ral Network', 2022 IEEE 2nd Mysore Sub Section International Conference (Mysu-ruCon) [Preprint]. doi:10.1109/mysurucon55714.2022.9972574.
[3] R Rumana, Reddygari Sandhya Rani, Mrs R. Prema, "A Review Paper on Sign Lan-guage Recognition for The Deaf and Dumb", International Journal of Engineering Re-search & Technology, volume 10 issue 10, 2021
[4] M. Hasan, T. H. Sajib and M. Dey, "A machine learning based approach for the detec-tion and recognition of Bangla sign language," 2016 International Conference on Medi-cal Engineering, Health Informatics and Technology (MediTec), Dhaka, Bangladesh, 2016, pp. 1-5, doi: 10.1109/MEDITEC.2016.7835387
[5] Papatsimouli, Maria & Kollias, Konstantinos & Lazaridis, Lazaros & Maraslidis, George S. & Michailidis, Heracles & Sarigiannidis, Panagiotis & Fragulis, George, "Real Time Sign Language Translation Systems: A review study". 1-4. 10.1109/MOCAST54814.2022.9837666
[6] Monika Chaturvedi, Bharti Kothari, Sandhya Singh, Vandana Rani Kujur, Aastha Ti-wari, "Sign Language Recognition", 2022 IJRTI | Volume 7, Issue 6 | ISSN: 2456-3315
[7] Amrita Thakur, Pujan Budhathoki, Sarmila Upreti, Shirish Shrestha, Subarna Shakya*, "Real Time Sign Language Recognition and Speech Generation", Journal of Innovative Image Processing (JIIP) (2020), Vol.02/ No. 02, Pages: 65-76

[8] A. Singh, A. Wadhawan, M. Rakhra, U. Mittal, A. A. Ahdal, and S. K. Jha, “Indian Sign Language recognition system for dynamic signs,” 2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Oct. 2022, doi: 10.1109/icrito56286.2022.9964940.
[9] C. Monikowski, “Language, cognition, and the brain: Insights from sign language re-search,” Studies in Second Language Acquisition, vol. 26, no. 3, p. 497–498, 2004.
[10] Anbarasi Rajamohan, Hemavathy R., Dhanalakshmi M., “Deaf-Mute Communication Interpreter”, International Journal of Scientific Engineering and Technology (ISSN: 2277-1581) Volume 2 Issue 5, pp: 336-341
[11] Farhan, Y. and Ait Madi, A. (2022) ‘Real-time dynamic sign recognition using Medi-aPipe’, 2022 IEEE 3rd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS) [Preprint]. doi:10.1109/icecocs55148.2022.9982822.
[12] R. C. Lee, K. K. Ng, C.-H. Chen, H. Lau, S. Chung, and T. Tsoi, “American sign lan-guage recognition and training method with recurrent neural network,” Expert Systems with Applications, vol. 167, p. 114403, 2021.
[13] Jamwal, A. et al. (2022) ‘Real-time conversion of American sign language to text with emotion using machine learning’, 2022 Sixth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC) [Preprint]. doi:10.1109/i-smac55078.2022.9987362.
[14] Kishore, K.K., Prudhvi, G. and Naveen, M. (2017) ‘Braille script to voice conversion’, 2017 International Conference on Computing Methodologies and Communication (ICCMC) [Preprint]. doi:10.1109/iccmc.2017.8282637.
[15] I. Culjak, D. Abram, T. Pribanic, H. Dzapo and M. Cifrek, "A brief introduction to OpenCV," 2012 Proceedings of the 35th International Convention MIPRO, Opatija, Croatia, 2012, pp. 1725-1730
[16] (No date a) NumPy. Available at: https://numpy.org/ (Accessed: 08 December 2023).
[17] Hand landmarks detection guide | mediapipe | Google for developers Google. Available at: https://developers.google.com/mediapipe/solutions/vision/hand_landmarker (Ac-cessed: 08 December 2023).
[18] OS - miscellaneous operating system interfaces (no date) Python documentation. Avail-able at: https://docs.python.org/3/library/os.html (Accessed: 08 December 2023).
[19] Carney, Michelle, Barron Webster, Irene Alvarado, Kyle Phillips, Noura Howell, Jordan Griffith, Jonas Jongejan, Amit Pitaru, and Alexander Chen. "Teachable machine: Ap-proachable Web-based tool for exploring machine learning classification." In Extended abstracts of the 2020 CHI conference on human factors in computing systems, pp. 1-8. 2020.
[20] React. Available at: https://react.dev/ (Accessed: 08 December 2023).
[21] Welcome to flask¶ Welcome to Flask - Flask Documentation (3.0.x). Available at: https://flask.palletsprojects.com/en/3.0.x/ (Accessed: 08 December 2023).
[22] GTTS (PyPI. Available at: https://pypi.org/project/gTTS/ (Accessed: 08 December 2023).
[23] D. Chaudhari, A. Dikshit, O. Pawar, Prof. Dr. D. Karia, and Asst. Prof. P. Shah, “Talk to Hands: A Sign Language Solution for Speech-Impaired People”, in International Conference on Computation of Artificial Intelligence & Machine Learning (Unpublished)