# Indian Sign Language to Text/Speech translation using YOLO Algorithm

CH Thiru Mohith
CSE
Presidency University
Bengaluru, India
thirumohithchitturi8@gmail.com

P Chakradhar Rao
CSE
Presidency University
Bengaluru, India
pantangichakradharrao@gmail.com

I Mohan Vamsi
CSE
Presidency University
Bengaluru, India
mohanindupuri2003@gmail.com

M Venkata Subash
CSE
Presidency University
Bengaluru, India
venkatasubashmandava@gmail.com

Dr. Smitha Patil
Assistant Professor
Presidency University
Bengaluru, India
smithapatil@presidencyuniversity.in

***Abstract*—**By utilizing cutting-edge computer vision and natural language processing techniques, the Sign Language Detection system is a creative initiative that helps people who use sign language communicate with others. This real-time solution recognizes and decodes hand motions that constitute the alphabet of sign language using a YOLO-based deep learning model. In order to facilitate smooth communication, the system converts detected motions into text, processes camera video input, and outputs voice commands. Furthermore, bidirectional interaction is supported by the system, which translates vocal inputs into sign language motions. It has a dynamic video feed, real-time annotation, idle-based word processing, and an intuitive UI thanks to Streamlit. For the hearing and speech-impaired groups, this project is a step toward increasing accessibility and inclusivity in communication.

**Keywords—Sign Language Detection,Gesture Recognition,Computer Vision,Real-Time Translation,Bidirectional Communication,Gesture-to-Text Conversion,Natural Language Processing**

## I. INTRODUCTION

A crucial communication tool for those who are hard of hearing or deaf is sign language. Nonetheless, a common obstacle to smooth communication between sign language users and non-users is the general lack of awareness about sign language. The Sign Language Detection project seeks to close this gap by developing a real-time system that can translate hand signals into voice and text outputs and vice versa. This initiative promotes diversity and enhances communication accessibility by fusing cutting-edge machine learning algorithms, computer vision, and natural language processing.

The YOLO (You Only Look Once) object identification model, a cutting-edge deep learning framework that recognizes hand motions that correspond to sign language letters, is the central component of the system. The application takes live video input from a webcam, recognizes movements in real time, and converts them into text. It is integrated with an intuitive UI created with Streamlit. After that, the text is transformed into audio outputs, facilitating effective and engaging conversation. By using users' speech inputs to generate equivalent sign language motions, the technology also enables bidirectional communication.

An important step in closing the gap between sign language users and non-users is this effort. It is a useful tool for inclusive communication because of its real-time capabilities, user-friendly interface, and bidirectional compatibility. In addition to addressing accessibility issues, this system promotes communication and understanding in a diverse society by leveraging developments in artificial intelligence and computer vision. For those who depend on sign language, it is a promising way to improve their quality of life.

## II. LITERATURE REVIEW

At the nexus of deep learning and computer vision, sign language recognition has emerged as a major research topic. The application of deep learning methods, specifically convolutional neural networks (CNNs) and YOLO (You Only Look Once) algorithms, for the recognition and translation of sign language motions has been the subject of numerous studies. For example, Bhuiyan et al. (2024) demonstrated notable advancements in bidirectional communication systems by combining YOLOv8 with natural language processing (NLP) to improve real-time gesture identification and translation [4]. Similar to this, Ahmadi et al. (2024) created a successful YOLO-based model for the recognition of Arabic sign language, with a noteworthy level of accuracy in real-time processing and gesture classification [2]. Due to its real-time gesture detection and classification capabilities, YOLO has been extensively investigated for use

in sign language detection. making it an ideal tool for sign language translation systems.

Deep convolutional networks have also been used to improve the accuracy of sign language recognition in addition to gesture identification. In order to achieve robust and dependable sign language classification, Kothadiya et al. (2022) presented "DeepSign," a deep learning-based system for sign language recognition that focuses on integrating CNNs with feature extraction techniques [7]. The ability of CNNs to comprehend both static and dynamic movements is demonstrated in this work, which is essential for developing effective and real-time sign language translation systems. Furthermore, Pigou et al. (2015) showed how CNNs may be used for continuous sign language recognition, demonstrating that deep learning models are capable of managing the complexity and diversity of human gestures in a variety of settings [11].These results highlight how useful CNNs are for identifying the wide variety of motions used in sign language communication.

The significance of combining several machine learning approaches to improve the functionality of sign language recognition systems has been the subject of numerous studies. By adding temporal context to static gesture identification, Rivera-Acosta et al. (2021) integrated YOLO with Long Short-Term Memory networks to enhance ASL translation, making it more appropriate for dynamic sign language movements [13]. The necessity for continuous recognition systems that adjust to ongoing gestures was also highlighted by Mocialov et al. (n.d.) in their discussion of continuous sign language recognition using deep learning models [9]. These developments pave the path for more accurate and adaptive communication solutions for the deaf and hard of hearing people by highlighting the expanding trend of merging numerous deep learning models to meet the difficulties of real-time sign language interpretation.

**SUMMARY OF LITERATURE SURVEY**.

The use of deep learning methods, specifically YOLO algorithms and convolutional neural networks (CNNs), for efficient gesture detection and translation is emphasized in the research on sign language recognition. While Kothadiya et al. (2022) and Pigou et al. (2015) show the effectiveness of CNNs in correctly detecting static and dynamic gestures, studies by Bhuiyan et al. (2024) and Ahmadi et al. (2024) emphasize the effectiveness of YOLO models in real-time sign language recognition. Furthermore, Rivera-Acosta et al. (2021) investigated the integration of methods such as LSTM networks, which shows promise for improving the recognition of continuous sign language motions. All things considered, these investigations highlight the expanding trend of integrating several deep learning models to raise the precision, speed, and versatility of systems that translate sign language.

## III. PROPOSED METHODOLOGY

### *A. Problem Statment*

Barriers to communication between sign language users and non-users provide serious obstacles to accessibility and inclusivity, frequently resulting in social exclusion and few possibilities for people who are deaf or hard of hearing. Real-time, scalable systems that provide smooth two-way communication are still absent, even with the availability of sign language interpreters and instructional initiatives. Effective communication in daily situations is hampered by the inability to convert gestures into voice or writing and vice versa. By creating a real-time sign language identification system that can transform sign motions into text and vocal outputs and voice inputs into sign language gestures, this project solves the issue and promotes understanding between people while removing barriers to communication.

### *B. Objectives*

**1.Real-Time Sign Language Recognition:** The main objective is to create a system that can identify sign language movements in real time. This entails teaching a deep learning model to recognize and decipher different sign language motions, particularly with the use of a computer vision technique (such as the YOLO method). Through training, the system will be able to recognize various static and dynamic motions with accuracy and translate them into text. People who are deaf or hard of hearing will be able to communicate effectively thanks to this, while people who may not be familiar with sign language will be able to understand their motions. To ensure the model's resilience and dependability in real-world situations, it must function in a variety of lighting conditions and with a range of hand sizes and shapes.

**2.Text-to-Sign Language Translation:** Enabling the conversion of typed or written text into comparable sign language motions is another crucial goal. This feature will meet the needs of those who want to communicate with persons who use sign language but do not know it. Every letter, word, or phrase in the input text will be represented by a specified gesture or animation. Deaf people and non-sign language users can communicate easily and interactively thanks to this two-way translation feature, which promotes understanding between the two groups.

**3.Improve Accessibility and inclusion:** By offering a technology that facilitates bidirectional communication between deaf, hard-of-hearing people and those who are unable to communicate using sign language, the project seeks to remove obstacles to communication and advance inclusion. For people who use sign language, the technology will improve information and service accessibility by offering real-time translation and feedback. Furthermore, it can provide a forum for communication between people with varying hearing levels without requiring an interpreter or a high level of sign language proficiency, thereby enhancing social inclusion.

4. **Optimize System Performance and Efficiency:** It is essential to optimize the system's performance in order to make sure that it is realistic and useful in real-world situations. This involves reducing latency during gesture

identification so that there are no noticeable lags in the real-time conversion of sign language to text and vice versa. The system needs to be precise and able to process a variety of movements from users, locations, and lighting situations. The system must be durable and very accurate in order to be dependable for daily use. Additionally, the system will be scalable, meaning it can handle several sign language systems and adjust to various regional or cultural gestures. This goal is essential to ensuring that the system is efficient and usable by a variety of users.

5. **Voice-to-Sign Language Translation:**
By incorporating a voice-to-sign language translation function, the project seeks to enable two-way communication. With this feature, users can speak into the system, and it will translate their words into movements in sign language. To analyze the spoken input, determine the matching sign language symbols, and animate the motions appropriately, the system will employ natural language processing (NLP) techniques. By ensuring good communication, this goal promotes two-way contact between sign language users and non-sign language speakers. Additionally, it expands the system's capabilities, making it more flexible and responsive to various communication requirements.

*C. Data Acquisition*

**1. Overview of the Indian Sign Language (ISL) Dataset:**
Images of different signs, including individual letter movements, words, and phrases used in Indian Sign Language, usually make up the Indian Sign Language (ISL) dataset. Both dynamic movements for lengthy sentences and static motions (such as letters or common words) are captured in these photographs. Sources such as Kaggle or datasets from organizations like the National Institute of Technology (NIT) are frequently accessible if you're utilizing a public ISL dataset. As an alternative, if the dataset is manually collected, it should include a wide variety of sign variations executed by several people to take into consideration variances in hand shapes, skin tones, signing pace, and environmental factors. This guarantees that the model may effectively generalize to many real-world situations.

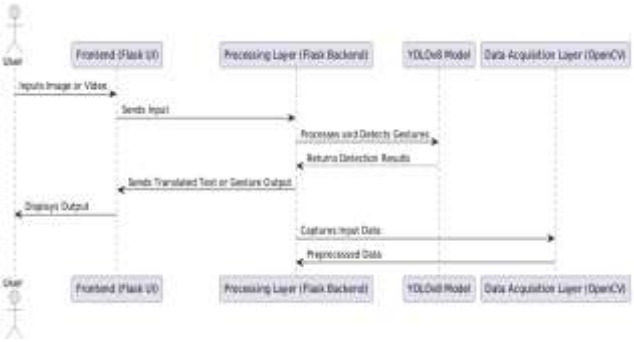
**2. Information Gathering and Labeling:**
To replicate the diversity of the actual world, it's crucial to make sure that motions are recorded in a variety of settings (such as with varying lighting, backgrounds, and hand orientations) when gathering the photos for your ISL dataset by hand. The hand gesture that corresponds to the related symbol should be clearly labeled on the photos. Labels could have individual letters like "A," "B," and "C" or words like "Hello," "Thank You," "Food," and "Water." The dataset as a whole should have these labels. The photographs can be annotated using programs like LabelImg or Labelbox, which guarantee that every gesture image has the appropriate label (class). In order to facilitate model training and evaluation, the annotations ought to adhere to a standardized framework, such as the YOLO format.

**3. Preprocessing Data**
To make sure the dataset is in a format appropriate for deep learning model training, data pretreatment is a crucial step. To meet the input size criteria of many pre-trained models, all photos should be downsized to a uniform resolution, such as 224x224 pixels. By dividing each pixel value by 255, it is also crucial to normalize the pixel values to the range [0, 1] or [-1, 1].The effectiveness of the model will be increased for ISL datasets by concentrating on the hand motions and removing extraneous background regions. The model can be made more resilient to various hand motion variations and ambient conditions by using data augmentation techniques including random rotations, horizontal/vertical flips, random cropping, and brightness or contrast adjustments.

**4. Data Division**
To adequately assess the model's performance, the preprocessed data must be divided into distinct subsets for training, validation, and testing. About 70–80% of the data is usually utilized for training, and the remaining 10-15% is used for validation in order to adjust the model's hyperparameters. To assess how effectively the trained model generalizes to new data, the remaining 10% to 15% is employed as a test set. For the model to correctly recognize each gesture, it is crucial to make sure that the dataset is divided at random while maintaining a fair distribution of gestures across all classes (letters, words, or phrases).

## IV. PROPOSED WORKFLOW

1. **Gathering and Preparing Data**
An Indian Sign Language dataset including pictures of still hand motions that depict letters or other regularly used signals is used in the research. The initial step is to annotate these photos in a YOLO-compatible format by creating bounding boxes and class labels using programs like LabelImg. Resizing photos to the YOLO model's input size (320x320 pixels, for example), standardizing pixel values for reliable feature extraction, and using data augmentation methods like flipping, rotation, and scaling are all examples of preprocessing. By taking these actions, the quality of the dataset is improved, the training instances are diversified, and the model's resilience is increased.




**2. Training of Models**
The effectiveness of the YOLOv8 model in real-time object recognition led to its selection. Setting up a.yaml file to specify the dataset format, class labels, and model parameters

is the first step in training. The annotated dataset is used to train the model, and its performance is maximized by optimizing hyperparameters such batch size, learning rate, and epochs. Validation data is utilized during training in order to prevent overfitting and track important metrics such as mean Average Precision (mAP). This stage is essential for creating a model that can reliably identify gestures in a variety of scenarios.

**3. Assessment of the Model**

Following training, the YOLO model's accuracy, precision, recall, and F1-score are assessed using a test dataset that has been put aside for that purpose. To examine the model's performance across several gesture classes, a confusion matrix is produced. These tests guarantee the model's dependability and preparedness for use in practical situations. Performance indicators show where the model might need to be improved.

**4. Real-Time Recognition of Gestures**

Real-time gesture detection is achieved by integrating a live video stream with the learned YOLO model. The YOLO model processes each video frame, identifying and categorizing hand motions and converting them into text. Low latency is ensured by the real-time display of the detected motions on the screen. By instantaneously translating their movements into text output, this feature enables deaf people to converse successfully.

**5. Development of User Interfaces**

Flask serves as a conduit between users and the ISL recognition technology, enabling the creation of an intuitive user interface. Among the many features of the interface is live gesture detection, which allows users to observe how their motions are converted into speech or text. Additionally, it allows regular users to input text and view the associated movements as images or animations by supporting text-to-sign conversion. Users can also contribute photographs for offline gesture detection using a file upload option. For users of various technical skill levels, the Flask-based user interface guarantees ease of use and accessibility.

**6.Converting Text to Speech (TTS):**

The matching letters or words are shown on the screen as the system recognizes and detects sign language gestures.At the same time, a text-to-speech engine is used to turn the identified text into speech. The text output is then vocalized by the system to facilitate communication for those who do not speak sign language.The speech output can be produced constantly by the system as it detects text or after it detects a word in its entirety.

**7.Translation from Voice to Sign Language:**

The user can use the voice-to-sign language translation functionality in addition to the sign-to-text and text-to-speech capabilities.The system translates spoken words into equivalent sign language movements when the user talks into the microphone.The voice input is interpreted by Natural Language Processing (NLP) algorithms, which then determine which sign language symbols match the spoken words.Sign language users can then see the words translated into hand gestures as the system animates and displays the gestures it has recognized.

## V. TECHNOLOGY

1. **You Only Look Once, or YOLO**

The foundation of the project is YOLO, more especially the YOLOv8 model, which is intended for real-time object identification. YOLO takes a one-stage approach, analyzing the full image in a single pass, in contrast to conventional object detection techniques that use multi-stage pipelines. Because of this, it is incredibly quick while still being highly accurate. With the help of its improved architecture, YOLOv8 can effectively recognize intricate hand gestures thanks to features like adaptive size augmentation and anchor-free detection. Because of its speed, sign language gestures may be translated into text almost instantly, which is essential for the system's real-time functionality.

2. **python**

The project's programming foundation is Python, which was selected due to its wide library ecosystem and versatility. Python makes it easier to implement the YOLO model and additional features with packages like Matplotlib for display, Pandas for dataset management, NumPy for numerical computations, and OpenCV for computer vision. It is perfect for team projects and quick development cycles because of its readability and simplicity. Additionally, Python makes it easy to integrate Flask, guaranteeing a single environment for both UI development and backend operations.

**3.Streamlit**

An open-source Python toolkit called Streamlit makes it possible to quickly create interactive web applications for data science and machine learning projects. By instantly converting Python scripts into fully working apps, it enables developers to quickly design web-based user interfaces. Because of Streamlit's incredibly user-friendly design, creating dynamic user interfaces using widgets like buttons, sliders, and text input fields is simple. Developers can concentrate on the backend processing and core logic while the library takes care of the front-end user experience automatically. Streamlit is an ideal option for developing the interactive elements of projects like sign language recognition systems since it is frequently used to create apps that visualize data, show machine learning models, and facilitate real-time interactions.

**4. OpenCV**

OpenCV is critical for managing video streams and carrying out necessary image preprocessing operations. The system preprocesses real-time video frames taken by the camera using OpenCV before forwarding them to YOLO for gesture recognition. It effectively manages operations like cropping, color space conversion, and scaling, guaranteeing that input data complies with YOLO's specifications. The system can

be adjusted to multiple hardware configurations because to OpenCV's compatibility with a wide range of camera devices.

**5.LabelImg**

LabelImg is used while preparing the dataset for annotation. With the help of this tool, users can create annotations that are compatible with YOLO by drawing bounding boxes around hand motions and assigning class labels. In order for the YOLO model to correctly recognize and categorize gestures during training, proper labeling is necessary. Even for enormous datasets, LabelImg's user-friendly interface makes annotation simple, making it a crucial tool for training data preparation.

6. **Pillow (PIL)**

The project uses Pillow, a robust image processing package, to manage picture alterations. Pillow transforms the video frame from the OpenCV format (BGR) into a Streamlit-compatible format (RGB) once OpenCV has finished processing it. Additionally, it creates borders, resizes photos, and gets them ready for the online interface. Pillow makes sure that the photos in this project are properly structured and prepared for display on the user interface, resulting in a seamless and aesthetically pleasing experience.

**7.Text to speech (TTS)**

The recognized text is transformed into spoken output by the Text-to-Speech (TTS) engine. Common libraries like pyttsx3 or gTTS (Google Text-to-Speech) can be incorporated into the project to vocalize the recognized words, even though the provided code makes no mention of the particular TTS engine. For users who might not understand sign language, the system becomes interactive when a word is finished since the TTS engine is activated and plays out the identified text. This feature improves accessibility by giving the user audio input.

**8.Time**

Controlling the timing of specific project operations requires the use of Python's time module. It helps determine when to register a new letter by keeping track of how long a sign is held. The idle time threshold is also controlled by the time module; if no gesture is detected for a predetermined amount of time, the system initiates the text-to-speech output for the current word. This guarantees that the system acts sensibly and reacts instantly to user input.

**9.Threading**

The threading library makes it possible to execute many activities at once, guaranteeing that the text-to-speech (TTS) and sign language identification features operate concurrently without interfering with one another. The TTS engine is utilized in this project to run in the background while the real-time sign language identification is carried out via threading. When a word is finished, the system can now instantaneously offer audio feedback without interfering with the continuing gesture detection process.

***Figures and Tables***

**System Architecture:**

The system's frontend is constructed with Streamlit, which offers an easy-to-use, interactive interface. By using buttons to start and stop the sign language identification process, the user engages with the system via Streamlit's online interface. The user is able to watch their gestures as the live video feed from their webcam is shown in real time. By continuously updating the interface with the recognized sign and associated text, Streamlit also controls user input for text and voice output, ensuring a seamless experience. Additionally, it manages feedback, including the identified letter or word, and shows it on the screen so the user may monitor their progress.

The system's backend is in charge of recording the webcam's video stream, processing each frame, and recognizing gestures in sign language. The OpenCV library, which enables effective video frame processing, is used to capture the video feed. To identify hand gestures, these frames are subsequently run via the Ultralytics YOLO framework-implemented YOLO (You Only Look Once) model. The model matches the identified hand signs to matched labels using pre-trained weights (best.pt) to identify individual letters in sign language. After a letter is identified, it is mapped to the letter from A to Z in sign language. The backend also manages time, including gesture hold duration and idle time thresholds, to guarantee that detected indications are processed and transformed into

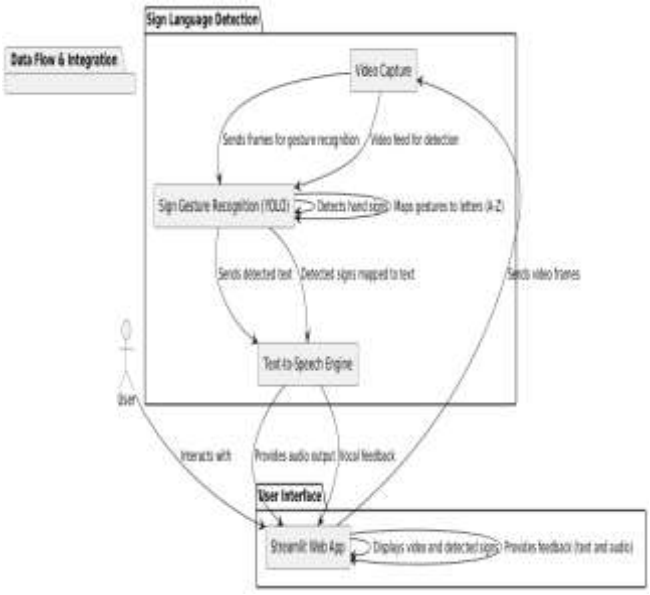



Fig 1 System Architecture

After a sign has been identified and translated into text, the system interprets the word and activates the Text-to-Speech (TTS) engine if no more signs are found during a predetermined amount of idle time. This engine provides the

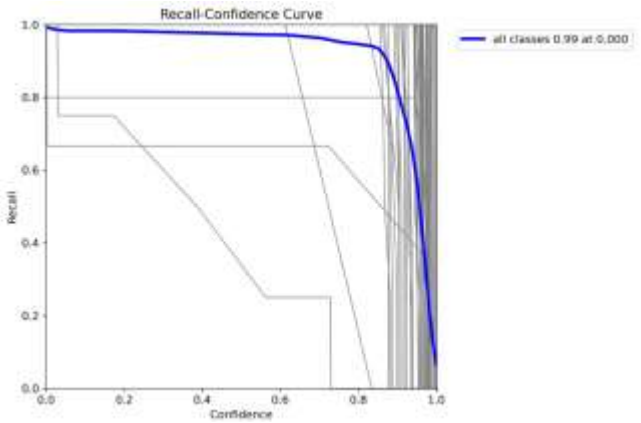
user with verbal feedback by turning the recognized word into audio. To make sure it doesn't interfere with the sign language recognition process, the TTS system uses threading to run in the background. The system can concurrently recognize signs, process the motions, and deliver uninterrupted speech response thanks to this asynchronous technique. The system's multi-threaded architecture enables efficient real-time sign language communication and recognition, giving the user an engaging experience.

**ACKNOWLEDGMENT**

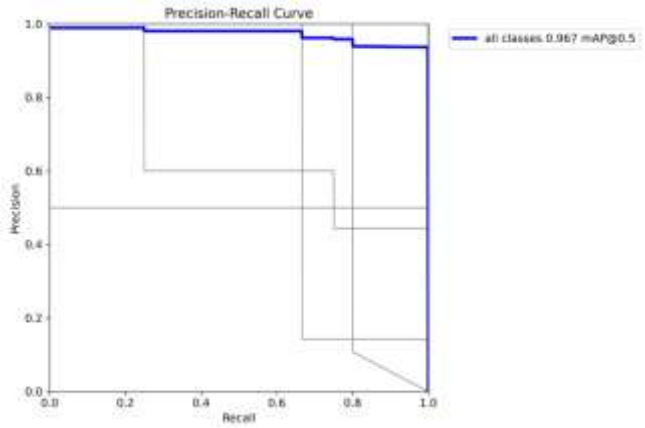
With deep appreciation, we would like to thank all the people and institutions that helped us finish this project successfully. We would like to express our sincere gratitude to our advisers and mentors for their crucial advice and assistance during the development process. The authors of the Indian Sign Language dataset, whose efforts have been crucial to the development and testing of our model, have our sincere gratitude. We are also grateful to the creators of Flask and YOLO, whose technologies served as the basis for our system. Finally, we thank our friends, family, and peers for their support and encouragement, whose faith in our vision has been a continual source of inspiration. Without their combined work and contributions, this project would not have been feasible.

## VI RESULTS AND DISCUSSION

Using a webcam feed, the Sign Language Detection system effectively illustrates real-time sign language detection. After being trained on a collection of sign language motions, the YOLO model can recognize individual letters in ASL with a high degree of accuracy. Every video frame is processed by the system, which recognizes hand gestures and associates them with the appropriate letter. Users can track their signs by viewing the identified letters in real-time on the Streamlit interface. Additionally, the system uses a text-to-speech engine to turn a string of letters into text and provide an audio output when the sequence is identified and held for a sufficient amount of time. This makes the procedure simple and interesting by enabling users to interact with the system using their sign language movements.




Although the system offers a reliable solution for detecting sign language, some difficulties were noted during testing. A number of variables, including illumination, hand gesture clarity, and hand position with respect to the camera, significantly affect how accurately sign language is recognized. When the hands were obscured or there was background noise in the video feed, the model occasionally had trouble identifying certain signs. Nevertheless, with more training data and model improvement, these problems can be lessened. Accessibility is improved by the backend's effective text-to-voice conversion, which produces precise and understandable speech output.




## VII CONCLUSION

In conclusion, the Sign Language Detection project effectively combines text conversion, audio output, and real-time gesture detection to help users who use sign language communicate. The system provides an intuitive and interactive experience by utilizing the YOLO model for hand gesture detection and integrating it with Streamlit for a smooth user interface. The method shows promise in bridging communication gaps for people with hearing impairments, despite obstacles such different lighting conditions and gesture clarity. This research has the potential to be a useful instrument for advancing accessibility and inclusivity with more enhancements, such as increasing the model's accuracy and broadening its gesture detection capabilities.

## REFERENCES

[1] *A Comprehensive Study on Deep Learning-Based Methods for Sign Language Recognition*. (2022). IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/9393618/

[2] Ahmadi, S. A., Mohammad, F., & Dawsari, H. A. (2024). Efficient YOLO based deep learning model for Arabic sign language recognition. *Research Square (Research Square)*. https://doi.org/10.21203/rs.3.rs-4006855/v1

[3] Alaftekin, M., Pacal, I., & Cicek, K. (2024). Real-time sign language recognition based on YOLO algorithm. *Neural Computing and Applications*, *36*(14), 7609–7624. https://doi.org/10.1007/s00521-024-09503-6

[4] Bhuiyan, H. J., Mozumder, M. F., Khan, M. R. I., Ahmed, M. S., & Nahim, N. Z. (2024, November 18). *Enhancing bidirectional sign language communication: integrating*

*YOLOV8 and NLP for Real-Time gesture recognition & Translation*. arXiv.org. https://arxiv.org/abs/2411.13597

**[5]***Deep convolutional neural networks for sign language recognition*. (2018, January 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/8316344/

**[6]***Deep learning methods for Indian sign language recognition*. (2020, November 9). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/9352194/

[7] Kothadiya, D., Bhatt, C., Sapariya, K., Patel, K., Gil-González, A., & Corchado, J. M. (2022). DeepSign: Sign language detection and recognition using Deep learning. *Electronics*, *11*(11), 1780. https://doi.org/10.3390/electronics11111780

[8] M, N. B., J, N. J. R., M, N. K., S, N. S., & M, N. B. (2021). Sign Language Translator Using YOLO Algorithm. *Advances in Parallel Computing*. https://doi.org/10.3233/apc210136

[9] Mocialov, B., Turner, G., Lohan, K., & Hastie, H. (n.d.). *Towards Continuous Sign Language Recognition with Deep Learning*. https://homepages.inf.ed.ac.uk/hhastie2/pubs/humanoids.pdf

[10] Oudah, M., Al-Naji, A., & Chahl, J. (2020). Hand Gesture Recognition Based on Computer Vision: A Review of Techniques. *Journal of Imaging*, *6*(8), 73. https://doi.org/10.3390/jimaging6080073

[11] Pigou, L., Dieleman, S., Kindermans, P., & Schrauwen, B. (2015). Sign language recognition using convolutional neural networks. In *Lecture notes in computer science* (pp. 572–578). https://doi.org/10.1007/978-3-319-16178-5_40

[12] Real time static and dynamic sign language recognition using deep learning. (2022). *Journal of Scientific & Industrial Research*, *81*(11). https://doi.org/10.56042/jsir.v81i11.52657

[13] Rivera-Acosta, M., Ruiz-Varela, J. M., Ortega-Cisneros, S., Rivera, J., Parra-Michel, R., & Mejia-Alvarez, P. (2021). Spelling correction Real-Time American Sign Language Alphabet translation System based on YOLO Network and LSTM. *Electronics*, *10*(9), 1035. https://doi.org/10.3390/electronics10091035

[14] *Sign language recognition using deep learning on custom processed static gesture images*. (2018, January 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/8537248/

**[15]**Wadhawan, A., & Kumar, P. (2020). Deep learning-based sign language recognition system for static signs. *Neural Computing and Applications*, *32*(12), 7957–7968. https://doi.org/10.1007/s00521-019-04691-y