Egyptian Journal of Artificial Intelligence
ISSN (Online): 2786-0213, ISSN (Print): 2786-0205
Volume 4, Issue 1, 2024, PP. x - xx

Bidirectional Translation in Sign Language

Mahmoud Y. Shams[1*], Alaa Hussien[1*], Hagar Ashraf[1*], Nada Nasser[2], Mariam Gabr[3], Hosam Mohamed[4], Atef Yasser[5]

[1,2,3,4,5]Faculty of Artificial Intelligence, Kafrelsheikh University, Kafrelsheikh,33516, Egypt.
*Corresponding author(s). E-mail(s): mahmoud.yasin@ai.kfs.edu.eg; alaa.ai 0267@ai.kfs.edu.eg; hagar.ai 0473@ai.kfs.edu.eg; Contributing authors: nada.ai 0462@ai.kfs.edu.eg; mariam.ai 0434@ai.kfs.edu.eg; hossam.ai 0296@ai.kfs.edu.eg; atef.ai 0332@ai.kfs.edu.eg;

Abstract: Sign language is often used by those who have hearing loss or speech problems to communicate. However, not everyone can understand the language. Automated sign language translation into text or the alphabet will let the deaf and the hearing impaired communicate with each other. In order to facilitate smooth communication, this research suggests a mobile application that allows two-way sign language translation. We use deep learning, more especially the YOLO v8 and YOLO v5 models, to translate and recognize sign language accurately**.** We find that YOLO v8 gets high mean average precision (MAP) scores of 95%, 96%, and 98% on the American-sign-language dataset and high accuracy, recall, and MAP scores of 98%, 97%, and 99% on the Sign Hands dataset. In a similar vein, YOLO v5 exhibits competitive performance with MAP scores of 94%, 95%, and 98% on the American sign-language dataset and 96.5%, 96.7%, and 98%, respectively, on the Sign Hands dataset. Two primary features of the system are available: 1) Text/Speech to ASL translates spoken or written English to text and then, using a 3D avatar, translates it into ASL animations; 2) ASL to Text/Speech translates ASL motions that are recorded by the camera into text or spoken English. The deaf and hearing communities may find this mobile application to be more accessible.

Keywords: American Sign Language, Blender, Avatar, 3D modeling, Android mobile application, YOLO v8 and v5.

1. Introduction

Globally, hearing loss affects more than 1.5 billion people. A further concern for nearly a billion children is hearing loss due to the improper usage of earphones and headphones. Children's growth, education, and mental and physical health are all adversely affected by these diseases. Children with hearing loss may face difficulties in communicating and delayed language development. Unfortunately, hearing loss is often not adequately accommodated in both public and private settings, which negatively impacts scholastic achievement and professional opportunities.

These days, sign language is highly significant. Humans can visually represent information to another person using hand motions as a kind of nonverbal communication. We call this sign language [1]. Sign language is frequently used by those who are deaf or have difficulty speaking. Since

learning sign language is not required, the majority of regular people do not acquire it. But people cannot just automatically be able to decipher sign language; they must first undergo a learning process if they ever need to. Not all individuals who are deaf can use the same sign language. Its unique characteristics and the challenge of collecting and understanding sign language make sign language detection a multifaceted task. Not all countries use sign language the same way, despite some clear similarities. A system capable of automatically identifying and deciphering signs, ranging from letters or text to photos or videos, will facilitate communication between the deaf and hard of hearing. Recently, deep learning—a subset of machine learning—has gained favor as a reliable and quick method for vision-based Sign Language Recognition (SLR). Thanks to its exceptional performance in image classification, object recognition, image capture, semantic segmentation, and human position prediction, deep learning has gained significant traction and credibility in the field of sign language studies.

Deep learning techniques also have the advantage of being able to examine large datasets, which is important for image processing. by contrasting the relevant hand model with the full captured and processed image. In vision-based SLR, deep learning seeks to yield a precise classification. American Sign Language (ASL)[2] is prioritized in many institutions, as is evident from a trend, because it is appropriate for model training. A deep learning method called "You Only Look Once" (YOLO) has been more well-liked lately for object recognition. The YOLO system was built using convolutional neural networks (CNNs), which offer rapid and precise object detection. In this study, we recognize real-world American Sign Language using Yolov5 and Yolov8. Fig. 1 shows the general functionality of our suggested system.

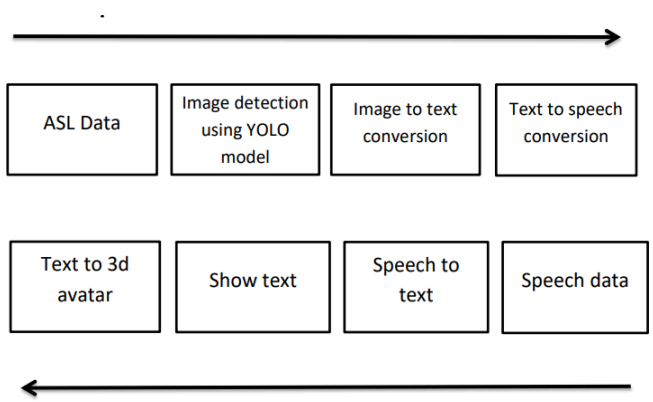


Fig. 1 Two-way communication between deaf and hearing people

2. Related work

People with hearing impairments use sign languages, which are defined as a systematic set of hand gestures with specific meanings, to communicate in everyday circumstances. Being visual languages, they communicate by moving their bodies, faces, and hands. Over 300 different sign languages are used worldwide. Even though there are many different sign languages, relatively few individuals are conversant in them, which makes it difficult for people with special needs to engage in casual conversation with everyone.SLR provides a means of communication for those who are not proficient in sign language. It translates a gesture into a widely spoken language, such as English. Many research projects are underway to convert American Sign Language to text. A noteworthy method makes use of MediaPipe, an open-source platform for creating ML pipelines with multiple modality support. Notably, MediaPipe provides a lightweight and effective detector for real-time applications in the form of the SSD-MobileNet-V2 [3] model.

2.1 Mediapipe

The situation of many disabled people can be improved by gesture-based communication recognition while still managing regular people; nonetheless, there are limitations to the use of gestures for communications. Therefore, in order to help people with hearing impairments learn and live better lives, a more advantageous methodology needs to be developed. Conventional methods like skeleton tracking, different color glove-based monitoring, and body component tracking have been heavily utilized in gesture recognition research. This problem has been resolved using a variety of methods, including CNN, SVM, and deep learning. AI computations for sign language recognition have been created in order to provide computer-based intelligence applications. Among them, machine learning

Using the MediaPipe[4] architecture, pipelines for time-series data in audio, video, and other forms are constructed. Google first made it available for real-time audio and video analysis on YouTube. With MediaPipe's 2019 public release, researchers and developers can now integrate and use this framework in their projects.

MediaPipe can perform efficiently, in contrast to the majority of high machine learning frameworks that demand a lot of processing resources. With the aid of the MediaPipe system, we can offer a model or computation for the application, and subsequently support the program by offering outcomes that are repeatable at various points in time. There are three main components to the MediaPipe system:

(1) an evaluation of the execution, and (2) a system for collecting data from the sensor (3) a collection of reusable components. A pipeline is a diagram that includes all of the components known as mini-computers. Each number cruncher is connected to channels that allow information to flow through them (Fig. 2). Designers can apply their essential modifications by removing or displaying number crunchers in the chart that are characterized by the client. The information stream graph is the result of adding devices and channels. Hand signal recognition using the MediaPipe architecture is a reliable addition to the high-loyalty hand and finger global positioning system.

The hand pose estimate provided by the Mediapipe module is accurate and dependable, allowing us to track and identify the movements and positions of both hands (Fig. 3) in real time. From the hand tracking module, we extract a total of 21 landmarks for each hand, logging their movements and spatial arrangement. These landmarks are essential parts of the latter stages of the sign language recognition paradigm.

Mediapipe hands use a coordinated machine learning pipeline comprising several collaborating models: (1) A palm recognizer analyzes the captured hand image; (2) A hand milestone model takes the handled image as input and returns the hand with resulting 3D central issues. (3) A motion recognition model that interprets the basic 3D hand movements and groups them into unique motion configurations. The process of identifying the palm model yields a clearly altered image of the palm that is subsequently sent off to the milestone model. This eliminates the need for information expansion, which is used in deep models to rotate, scale, and flip images. The process of identifying hands is time-consuming and problematic due to the need to work with different hand sizes, thresholding, and image handling.

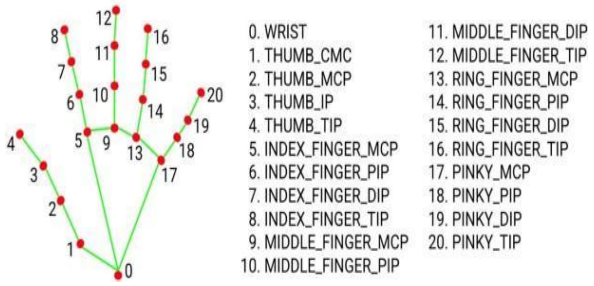
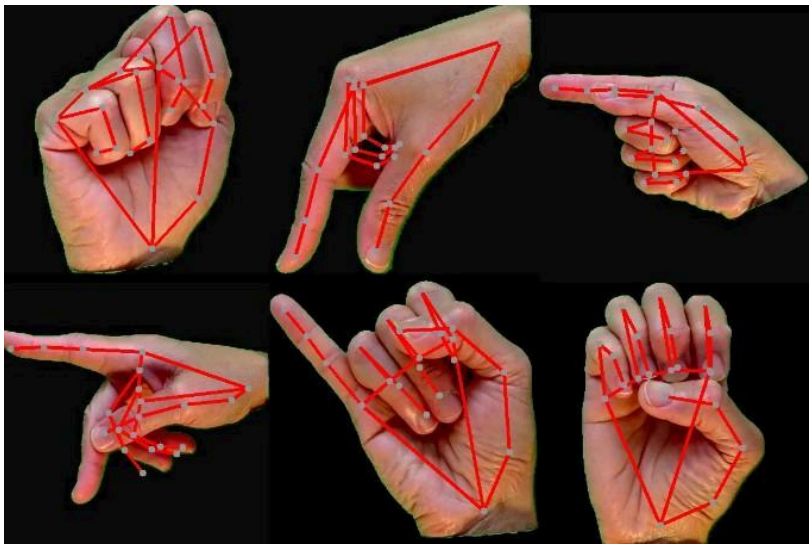


Fig. 2 Mediapipe Hand Tracking Module Landmarks



.

Fig. 3 sign language detection using mediapipe

2.2Mobilenet SSD V2

Popular object detection model MobileNet V2 includes two deep learning architectures: SSD (single shot detector) and MobileNet V2. For object detection, MobileNet V2 is renowned for its quickness, effectiveness, and high accuracy. SSD is an object detection framework that has a high degree of efficiency and accuracy in detecting multiple items in a picture. The SSD Image size of MobileNet V2 is 320x320. Figure 4 illustrates its design.

A Single Shot Detector (SSD) object detection network and a MobileNet V2 backbone network make up the two primary parts of the SSD MobileNet V2 architecture. Lightweight convolutional neural networks like the MobileNet V2 backbone network are made to function well on mobile devices with constrained processing power. The backbone network consists of 53 convolutional layers, including depth-wise separable convolutions, which enable efficient feature extraction with fewer parameters. Using the input image, features are extracted from the MobileNet V2 backbone network, which has undergone pre-training on a substantial photo dataset.

By forecasting the bounding boxes and class labels for every object in the image, the SSD object detection network is able to identify the objects. To make the model map between the alphabets in this instance, label mappings were made, such as: Since ID for the alphabet "A" was initialized to 0, the model maps to the specified ID—in this case, "0"—when it is tested and trained on images of the alphabet. This is how the model is able to categorize photos for all alphabet sets. Several detection layers in the SSD network let it to recognize objects with a range of scales and aspect ratios.

The feature extraction layers of the MobileNet V2 backbone network are linked to the detection layers, allowing them to extract features from the input image. The Convolutional Layer is one of the several layers that make up the SSD network. Its function is to extract features from the input images. Activation Layer: Non-linearity is added to the network using the activation layers. Layer of Prediction: For every image in that item, the bounding box coordinates and class labels are predicted using the prediction layers.

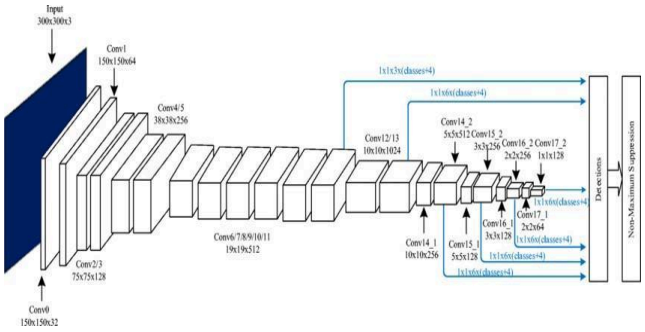


Fig. 4 SSD MobileNet V2 architecture

3.Methodology

3.1 YOLOv5

For object detection tasks, YOLOv5 [5] is a deep learning-based architecture. YOLOv5 requires less processing power and provides a convincing balance between speed and accuracy when compared to several state-of-the-art models. Notably, Glenn Jocher, an independent software developer, released YOLOv5 as the first open-source implementation.

of a research paper that serves as support [1]. The project is listed as "ongoing development" and is maintained on GitHub (https://github.com/ultralytics/yolov5). YOLOv5, which was created in Python, places a high priority on simplicity of setup and integration, especially for

Internet of Things (IoT) device deployment. In addition, compared to the Dark-net framework utilized in previous YOLO versions, the PyTorch platform offers a more vibrant development community. YOLOv5 leverages the Path Aggregation Network (PANet) to improve information flow inside the model and CSPDarknet53 as its backbone for feature extraction to gain its speed advantage (see Fig 5).

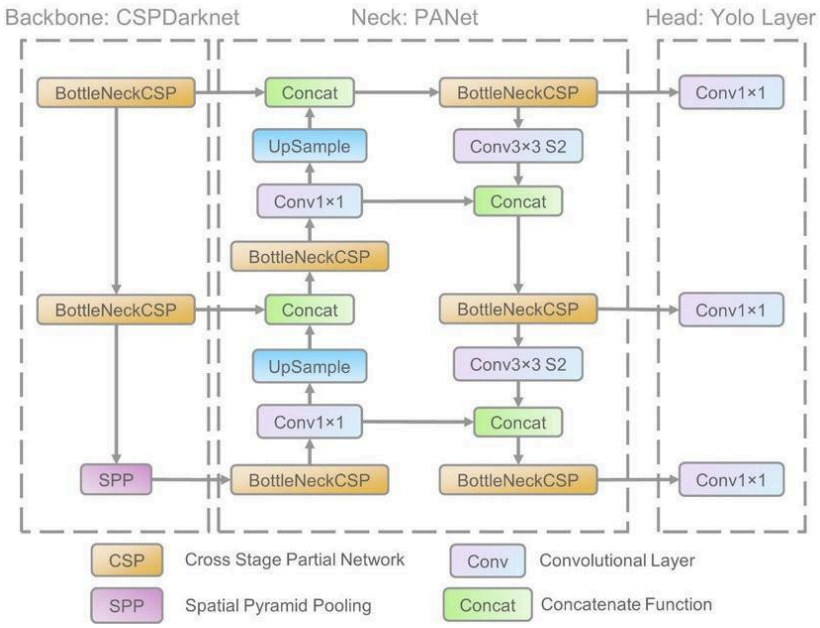


Fig. 5 YOLOv5 architecture

The usage of YOLOv5 is justified for the following reasons:
SOTA elements including an activation function, hyperparameter, data augmentation strategy, and an understandable user manual are included in the YOLOv5.
Even with limited resources, the model's straightforward architecture allows for computational convenience during education.
Yolov5 is perfect for embedded and cellular applications because of its compact size and light weight.

3.2 YOLOv8

Glenn Jocher [6] released YOLOv8, one of the most recent versions of the YOLO object detection model family, in January 2023. This state-of-the-art computer vision model has novel capabilities including mosaic data augmentation and anchor-free recognition and is built for real-time applications. In comparison to earlier iterations, a Python package expedites the development process and a user-friendly command-line interface (CLI) makes model setup simpler. (Source: ultralytics/ultralytics on GitHub).

To use YOLOv8, an image must be analyzed. It then forecasts class probabilities—the possibility that an object belongs to a particular class—as well as bounding boxes, which are the regions around objects, simultaneously. The image is split into a grid of uniformly sized cells in order to accomplish this. The bounding boxes of each cell are predicted by YOLOv8 to have confidence scores that represent the likelihood that an object will exist inside that box.
YOLOv8 is an effective tool for real-time object detection tasks because of its speed and precision. Hand gesture identification in video outlines is one possible use.
Any object detection system's accuracy, including YOLOv8 systems, depends on a number of variables. These variables include the model's architecture, the quantity and quality of training data used, and hyperparameter tuning (adjusting the model's configuration parameters).

3.3 Training process

The medium versions of YOLO v5 and YOLO v8 were trained for sign language recognition in this study.

Key Training Parameters.
Image size: To guarantee consistency for the model's input, we shrunk every image in the training dataset to a standard size of 640x640 pixels.
batch size: sixteen were employed in the training.
The quantity of epochs was 150 for YOLO v5 and 200 for YOLO v8.
for one hundred epochs.
We found that our YOLO v5 and YOLO v8 models outperformed methods that used MediaPipe and the SSD-MobileNet-V2 model. Yolo's abundance of features allows it to overcome the shortcomings of other object detection algorithms.
1. Single Shot Detection: Yolo is a single-stage object detection model that

processes the entire image at once to predict bounding boxes and class probabilities for items. Its efficiency allows it to be applied to real-time tasks such as hand detection.

2. Robustness to Size Variations: Yolo is designed to identify objects at different sizes within a picture. By employing a grid-based technique to predict bounding boxes and class probabilities over the whole image, it manages item size variations well.

3. End-to-End Training: Yolo receives end-to-end training, which enables it to simultaneously learn how to optimize the localization and classification tasks. When recognizing hands of various sizes and orientations, this might enhance robustness and generalization.

4 Dataset

The dataset that serves as the input for the sign language recognition system consists solely of sign hands in American Sign Language (ASL)[7], while additional data includes words and characters in ASL [8].

4.1Sign hands:

has 26 classes ranging from A to Z (Fig. 7), with data divided into test, validation, and train sets. Table 1 provides data specifications.

Table 1 data description

specification	value
Numbers of images	1815
Train data	1271(70%)
Valid data	363(20%)
Test data	181(10%)
Number of classes	26
Number of images per class	70

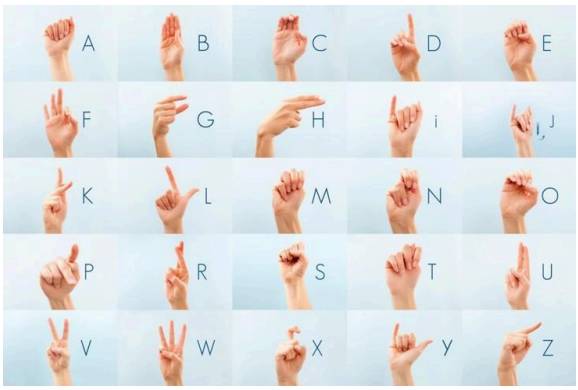


Fig. 7 Sign hands dataset

4.2 American-sign-language (ASL):

The films in this dataset were divided into frames (Fig. 8). Each frame had 106 classes—26 of which were characters and 80 of which were words—and was annotated. Three sets of data exist here: train, validation, and test. Table 2 provides data specifications.

Table 2 data description

specification	value
Numbers of images	23343
Train data	20630(88%)
Valid data	1768(8%)
Test data	945(4%)
Number of classes	106

5 Results

Two distinct datasets that were divided into training, testing, and validation were employed. Yolo v5 achieved 96.7% recall, 96.5% precision, and 98% mean average precision (MAP) for the character-only dataset, while Yolo v8 achieved 97% recall, 98% precision, and 99% mean average precision (Table 3).

Yolo v5 and Yolo v8 yielded 95% recall, 95% precision, and 98% mean average precision and 96% recall, 95% precision, and 98% mean average precision, respectively, in the second dataset (Table 4).

Figure 13. displays the second dataset's v5 and v8 confusion matrices. In terms of bounding box and classification loss, Yolo v5 has dropped more

quickly than Yolo v8 (Fig 14). When it comes to sign language recognition, Yolo v8 is quicker and more precise than Yolo v5.

Table 3 results of sign hands dataset

model type	precision	recall	MAP	Box loss	Classification loss
YOLO v5	96.5%	96.7 %	98%	0.01	0.009
YOLO v8	98%	97%	99%	0.06	0.09

Table 4 results of American-sign-language (ASL) dataset

model type	precision	recall	MAP	Box loss	Classification loss
YOLO v5	94%	95 %	98%	0.01103	0.00135
YOLO v8	95%	96 %	98%	0.02	0.066

6 3D

One exciting frontier in sign language translation technology is the use of 3D avatars. While traditional systems may rely on text or 2D animations, 3D avatars offer a more natural and expressive way to convey signs. This section will explore the potential of 3D sign language translation as shown in Fig.9

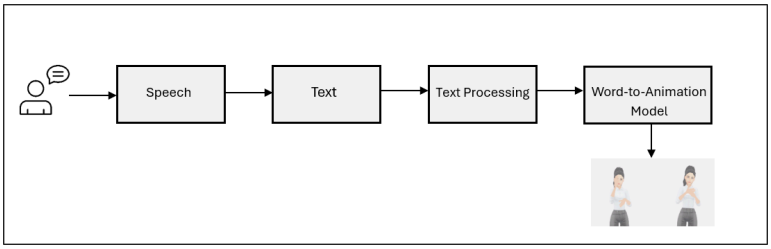


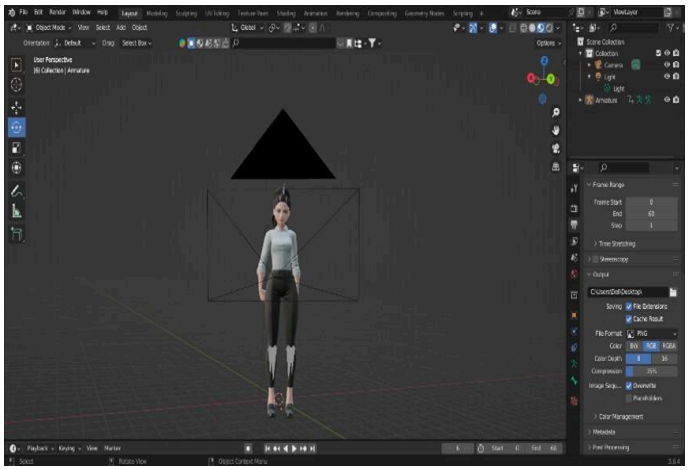
Fig. 9 Generating 3D Avatar Animations from Speech Input.

6.1 Avatar Creation

Blender is an open-source tool for creating 3D animations[9], It is used to execute Python scripts for different functionalities. [10].

Modeling: This involves creating the basic structure of the avatar using modes like pose mode, object mode, and edit mode. Starting with the fundamental structure, such as modeling the fingers before the palm in a human body, the avatar's form is gradually built up (Fig. 10). This process includes shaping meshes, curves, and other elements to form the desired appearance.

Fig. 10 Avatar Modelling in Blender



Rigging: Once the model is created, bones are added and attached to the structure (Fig. 11). This creates a skeletal framework, which is known as rigging. In rigging, a wireframe surrounds the body to define its boundaries. For example, in a 3D avatar, finger bones would be connected to the mesh representing the hand. Rigging also involves applying constraints to the bones, such as inverse kinematics constraints and bone constraints. Additionally, specific bones may have kinematic constraints applied to them. Euler transforms are commonly used for bone rotation in this stage.

Fig. 11 Avatar Rigging in Blender

Animation: encompasses the dynamic alteration of an object's position or form over a duration. This dynamic transformation can be realized through various techniques, including relocating the object as a unified entity by altering its position, reshaping it by manipulating its individual vertices or control points, and imple- menting inherited animations where an object's movement is influenced by another object's motion. Keyframes serve as pivotal markers for storing property values at specific time instances. In this system, keyframes are established at intervals of 25 frames within an animation timeline comprising a total of 60 frames. The pace of the animation output can be adjusted to suit desired speed parameters.
Rendering: involves the translation of a 3D scene into a sequence of 2D images,
culminating in the creation of a video. In Blender, a diverse array of video formats such as MPEG-1, AVI, MPEG-2, and H.264 (MP4) are supported. Following the animation process, data is organized and stored in individual frames. Rendering encompasses the execution of instructions to assemble these frames into a coherent visual narrative. The resulting animation is subsequently converted into the designated video format. In the context of this system, the rendered video output is formatted as FFmpeg Video and the Video Codec is H.264 with medium quality and good encoding speed.

6,.2 Animation script and Time

Blender, equipped with a Python script editor, is utilized in this context to gener- ate animation. There are start and stop keyframes given to every word in the final text translator block's output file. Initially, the avatar is positioned in the default T-pose(Fig. 12).

Fig. 12 T-pose

The beginning and ending positions corresponding to the sign are fetched from memory in respect to the word being processed in the loop during the start and end keyframes. The bones are then given the X, Y, and Z coordinates.
Rendering and creating videos are the two most time-consuming parts of the sign language translator. The command line displays the amount of time needed, which we measured. There is a difference in the amount of time required for text translation between translating longer and shorter sentences:

- Rendering a single word like "sad" takes 70 seconds.
- Generating a 70-frame video, such as" I love you", requires 120 seconds.

The rendering time in the sign language translator is primarily influenced by the following factors:
1.Complexity of Animation: The level of detail in the animation directly affects rendering time. More complex animations with intricate movements and effects typically require longer to render.
2.Number of Frames: The total number of frames in the animation is a crucial determinant of rendering time. Higher frame counts result in longer rendering durations.
3.Hardware Resources: The processing power of the hardware, including CPU and GPU capabilities, as well as available RAM, significantly impacts rendering speed. More powerful hardware generally leads to faster rendering times.
4.Render Settings: Parameters such as resolution, quality settings, and ren- dering techniques influence rendering time. Higher resolutions and quality settings typically require more processing time.
5.Software Optimization: The efficiency and optimization of the rendering soft- ware also play a vital role. Well-optimized rendering engines can substantially reduce rendering times compared to less

optimized ones.

By optimizing these key factors, developers can effectively manage and minimize rendering times in the sign language translator. To illustrate the 3D Avatar's capabilities.

7.Figures

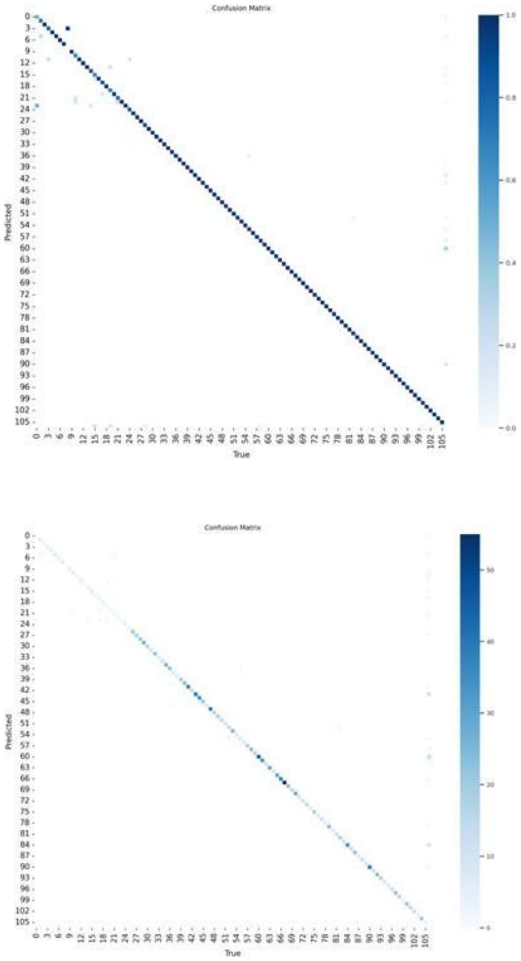


Fig 13. Confusion matrix for YOLOv5(above) and YOLOv8(bleu)

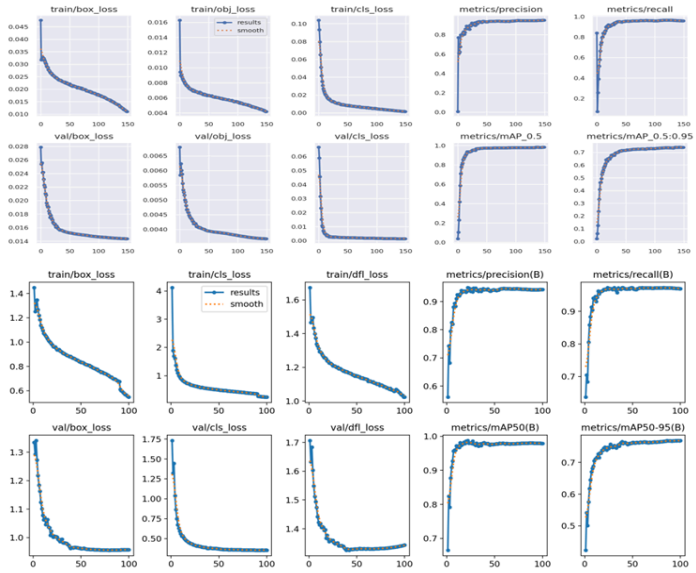


Fig 14. Results and loss reduction for YOLOv5(above) and YOLOv8(below)

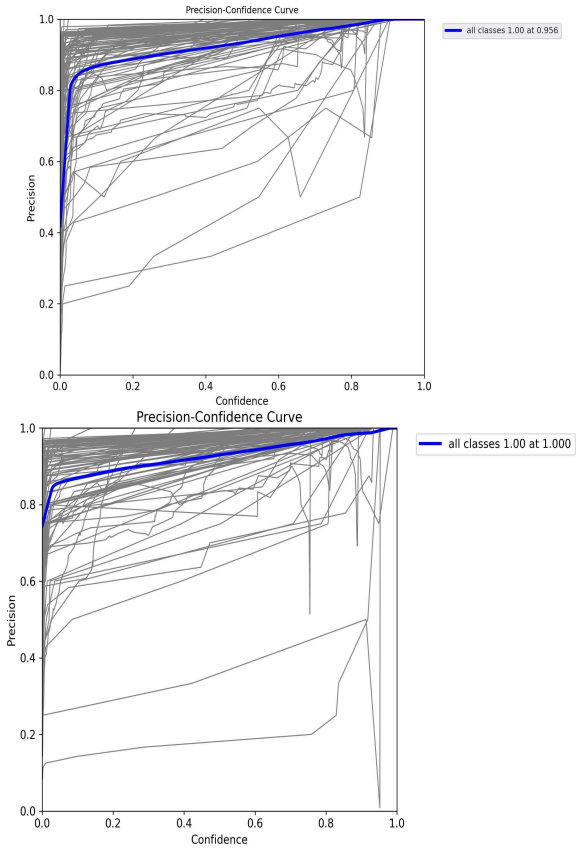


Fig 15. Precision curve for YOLOv5(above) and YOLOv8(below)

8.Conclusion

This research aims to address the communication barrier between deaf and hearing individuals by developing a vision-based American Sign Language (ASL) recognition system using YOLO architecture. The research successfully explored the development of a mobile application for two-way communication between deaf/mute and hearing individuals. The application leverages YOLO v8 and YOLO v5 deep learning architectures to achieve sign language recognition and translation.

Key Findings.

The study demonstrates the effectiveness of YOLO architectures for sign language recognition. YOLO v8 achieved superior performance compared to YOLO v5 on both datasets, reaching a maximum precision of 98%, recall of 97%, and mean Average Precision (MAP) of 99% for character recognition.
The application exhibits promising results for recognizing both individual characters and words in American Sign Language (ASL).

Acknowledgements. We would like to express our sincere gratitude to Eng. Abdel Maula Youssef for his invaluable support throughout this research project. His contributions significantly enhanced the quality of this research.

Future Work

Expanding the ASL sign vocabulary within the application for broader communication capabilities.
Embedding Hardware for Enhanced Accessibility. This might include wear-able devices like gloves or sensors that can capture sign language data with greater precision and potentially improve recognition accuracy in various lighting conditions or hand positions.
Multilingual Support for Global Accessibility. To broaden its reach and foster communication across international borders, future development will involve incorporating additional sign languages.
Integration of Augmented Reality (AR) Technology. Imagine a user holding their phone up and seeing corresponding ASL signs displayed as virtual annotations alongside a spoken or written sentence. This immersive approach could revolutionize sign language education and communication for both deaf/mute and hearing users

References

[1] Kyle, Jim G., James Kyle, and Bencie Woll. Sign language: The study of deaf people and their language. Cambridge university press, 1988.

[2] Padden, Carol, and Claire Ramsey. "American Sign Language and reading ability in deaf children." Language acquisition by eye 1 (2000): 65-89.

[3] Saiful Bahri, Iffah Zulaikha, et al. "Interpretation of Bahasa Isyarat Malaysia (BIM) Using SSD-MobileNet-V2 FPNLite and COCO mAP." Information 14.6 (2023): 319.

[4] Harris, Moh, and Ali Suryaperdana Agoes. "Applying hand gesture recognition for user guide application using MediaPipe." In 2nd International Seminar of Science and Applied Technology (ISSAT 2021), pp. 101-108. Atlantis Press, 2021.

[5] Wu, Wentong, et al. "Application of local fully Convolutional Neural Network combined with YOLO v5 algorithm in small target detection of remote sensing image." PloS one 16.10 (2021): e0259283.

[6] Terven, Juan, Diana-Margarita C´ordova-Esparza, and Julio-Alejandro Romero- Gonz´alez. "A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas." Machine Learning and Knowledge Extraction 5.4 (2023): 1680-1716.

[7] https://universe.roboflow.com/atuzim/american sign language-sgkct

[8] https://universe.roboflow.com/majorproject-25tao/american-sign-language-v36cz

[9] Blender 4.1 Reference Manual. https://docs.blender.org/manual/en/latest/

[10] Blender Foundation, "Game Engine - Blender Manual," Blender Foundation, [Online]. Available: https://docs.blender.org/manual/en/dev/game engine/index. html

[11] Flavell, Lance. Beginning blender: open source 3d modeling, animation, and game design. Apress, 2011.

[12] Das Chakladar, Debashis, Pradeep Kumar,

Shubham Mandal, Partha Pratim Roy, Masakazu Iwamura, and Byung-Gyu Kim. 2021. ”3D Avatar Approach for Continuous Sign Movement Using Speech/Text” Applied Sciences 11, no. 8: 3439. https://doi.org/10.3390/app11083439

[13] Petkar, Tanmay, Tanay Patil, Ashwini Wadhankar, Vaishnavi Chandore, Vaish- navi Umate, and Dhanshri Hingnekar. ”Real Time Sign Language Recognition System for Hearing and Speech Impaired People.” Int J Res Appl Sci Eng Technol 10, no. 4 (2022): 2261-2267.

[14] Adamo-Villani, Nicoletta. ”3d rendering of american sign language finger-spelling: a comparative study of two animation techniques.” International journal of human and social sciences 3, no. 4 (2008): 24.

[15] Alaftekin, Melek, Ishak Pacal, and Kenan Cicek. ”Real-time sign language recog- nition based on YOLO algorithm.” Neural Computing and Applications (2024): 1-16.

[16] Patel, M. (2023). American sign language detection (Doctoral dissertation, California State University, Northridge).

[17] Halder, Arpita, and Akshit Tayade. ”Real-time vernacular sign language recog- nition using mediapipe and machine learning.” Journal homepage: www. ijrpr. com ISSN 2582 (2021): 7421.

[18] 1.Elakkiya R (2021) Machine learning based sign language recognition: a review and its research frontier. J Ambient Intell Humaniz Comput 12:7205–7224

[19] Vidhyasagar, B. S., An Sakthi Lakshmanan, M. K. Abishek, and Sivakumar Kalimuthu. ”Video Captioning Based on Sign Language Using YOLOV8 Model.” In IFIP International Internet of Things Conference, pp. 306-315. Cham: Springer Nature Switzerland, 2023.

[20] Lui, Michael Stephen, and Fitri Utaminingrum. ”A Comparative Study of YOLOv5 models on American Sign Language Dataset.” Proceedings of the 7th International Conference on Sustainable Information Engineering and Technology. 2022.

[21] Bora, Jyotishman, et al. ”Real-time assamese sign language recognition using mediapipe and deep learning.” Procedia Computer Science 218 (2023): 1384-1393.

[22] Shams, Mahmoud Y., et al. ”Food Item Recognition and Calories Estima- tion Using YOLOv5.” International Conference on Computer Communication Technologies. Singapore: Springer Nature Singapore, 2023.

[23] Shams, Mahmoud Y., Omar M. Elzeki, and Hanaa Salem Marie. ”Towards 3D vir- tual dressing room based user-friendly metaverse strategy.” The Future of Metaverse in the Virtual Era and Physical World. Cham: Springer International Publishing, 2023. 27-42.