International Journal of Advances in Engineering and Management (IJAEM)
Volume 6, Issue 11 Nov. 2024, pp: 37-45 www.ijaem.net ISSN: 2395-5252

Sign language recognition using convolutional neural network

Dr.Tejaswi Potluri[1], Arshiya Mehnaz [2], T Amrutha Varsha [3], Taneti Priyadharshini [4], Vaddepalli Bhavana[5]

Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering &Technology
Vignana Jyothi Nagar,Pragathi Nagar, Nizampet,
Hyderabad, Telangana 500090

Date of Submission: 01-11-2024 Date of Acceptance: 10-11-2024

ABSTRACT: Sign language detection plays a vital role in enhancing accessibility, communication, and inclusion for individuals who are deaf or hard of hearing. It is an essential tool that facilitates smooth communication in various environments, such as schools, workplaces, healthcare settings, and daily interactions. By accurately identifying and interpreting sign language gestures, this technology helps bridge communication barriers between sign language users and those unfamiliar with it, ensuring equal access to information and services. In education, this technology supports language learning, literacy, and academic achievement for deaf students. In healthcare, reliable sign language detection ensures that medical information and consultations are conveyed clearly, improving care quality for deaf patients. In the workplace, it promotes inclusivity by enabling effective communication between deaf employees, their colleagues, and clients, fostering equal opportunities and professional growth. Additionally, during emergencies, this technology can save lives by facilitating swift and precise communication between emergency responders and deaf individuals. When incorporated into assistive technologies, sign language detection enables deaf individuals to independently handle everyday tasks, such as accessing digital content, using smart devices, and engaging in online interactions.

KEYWORDS: Convolutional neural networks, Image processing, Feature extraction.

I. INTRODUCTION

The Sign To Text project is designed to address the communication needs of the global deaf and hard of hearing community. With around 70 million people worldwide relying on more than 300 different sign languages as their main form of communication, this initiative represents a significant step forward in improving communication accessibility. It aims to bridge the gap between individuals who are hearing-impaired and those who are not, utilizing advanced image recognition technology and deep learning algorithms to interpret sign language gestures in real-time through a camera, providing effortless communication. Sign language detection, specifically through convolutional neural networks (CNNs), marks a major breakthrough in machine learning for accessibility and communication. CNNs, a deep learning model specialized for image recognition, are used in this project to accurately interpret and classify sign language gestures. By developing this technology, we aim to tackle the communication barriers that deaf or hard of hearing individuals face daily. This introduction provides an overview of the methodology, goals, and potential benefits of using CNNs to recognize and interpret sign language, enhancing accessibility, fostering inclusivity, and enabling effective communication in various settings. The development of deep learning techniques, particularly CNNs, has transformed the ability to automatically detect and classify visual patterns in sign language gestures. CNNs are highly effective in identifying spatial relationships and hierarchical features in images, making them ideal for recognizing the complex hand shapes, movements, and facial expressions characteristic of sign language. This project seeks to utilize CNNs to build a reliable sign language detection system. By training the network on a diverse dataset of sign language images and videos, the system will learn to accurately differentiate between various signs for letters, words, and phrases across multiple sign languages. Through extensive testing and validation, we aim to achieve high levels of

International Journal of Advances in Engineering and Management (IJAEM)
Volume 6, Issue 11 Nov. 2024, pp: 37-45 www.ijaem.net ISSN: 2395-5252

accuracy and reliability in real-time sign language recognition. The potential impact of this project is substantial, with the ability to significantly enhance communication for the deaf and hard of hearing community..

II. LITERATURE SURVEY

It consists of a comprehensive examination and synthesis of existing academic papers, articles, and relevant publications that pertain to the specific domain or problem being addressed. By delving into prior work, researchers gain a deep understanding of the historical context, theoretical foundations, and practical applications related to their area of interest within ML. [1-20] This exploration not only helps in identifying key concepts, methodologies, and algorithms that have proven effective but also highlights gaps in knowledge or unresolved issues. Moreover, the literature review serves as a guide for selecting appropriate ML techniques, models, and datasets based on their performance in similar contexts. The reviewed papers present a broad spectrum of research in sign language recognition and translation, showcasing advancements in various methodologies and technologies. A significant number of studies focus on leveraging machine learning techniques such as Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and Recurrent Neural Networks (RNNs) to improve sign language translation systems. For instance, Bantupalli and Xie [1] explored American Sign Language (ASL) recognition using these methods, while Katoch et al. [6] applied CNNs combined with SURF features and SVMs for Indian Sign Language (ISL) recognition. The research indicates that deep learning approaches, particularly CNNs, have become dominant due to their superior ability to extract high-level features from visual data. Several papers delve into specific techniques and their effectiveness. Ahmad et al. [2] and Agre et al. [8] highlighted real-time translation systems, with Ahmad's work focusing on the BISINDO dataset and Agre's study on text and speech conversion. Similarly, Akano and Olamiti [3] and Lang et al. [4] examined the conversion of sign language to text and speech using machine learning and Kinect, respectively. These studies illustrate the progression from static image recognition to dynamic, real-time translation systems, addressing various challenges such as gesture variability and computational efficiency. Challenges in the field include dealing with dynamic gestures, variations in facial features, and the need for more comprehensive datasets. Papers such as those by De Souza and Pizzolato [7] and Harini et al. [14] discuss the use of Hidden Conditional Random Fields (HCRFs) and hybrid models to tackle these issues. The need for larger and more diverse datasets is emphasized by Ben Haj Amor et al. [5], who reviewed electromyographic signals for sign language recognition, suggesting that current datasets may not fully represent the diverse sign language variations. Future directions in sign language recognition and translation research include real-time systems, multilingual support, and integration with emerging technologies. Papers like those by Liang et al. [16] explore these areas, focusing on the use of deep learning and hybrid approaches to enhance translation capabilities and accessibility. The ongoing research aims to improve the practical application of sign language systems by addressing the remaining challenges and integrating advanced technologies for better real-time performance and broader accessibility.

III. METHODOLOGY

A. Dataset Description

In our project, an existing dataset containing 2515 images representing 36 classes (A to Z and 0 to 9) was used. This dataset ensures comprehensive coverage of the American Sign Language (ASL) alphabet and numerals. Each class corresponds to a specific letter or digit, facilitating accurate gesture recognition.

B. Data Preprocessing & Feature Extraction

In this project, preprocessing focused on resizing images and splitting the dataset for training and validation. The images were resized to a resolution of 224 x 224 pixels using the Pillow library (**PIL**), ensuring consistency with the input requirements of the Convolutional Neural Network (CNN). The resizing process maintained the dataset's structure, saving the processed images in a new directory. After resizing, the dataset was split into training and validation sets using the **splitfolders** library, with 70% of the data allocated for training and 30% for validation.

C. CNN Architecture - Training and Testing

The CNN model was built using the Keras library and includes several key layers, each contributing to the overall performance of the model. Here's a detailed explanation of the architecture, including the formulas used in the layers:

Convolutional Layers :

1st Convolutional Layer: The input to this layer consists of images with dimensions

International Journal of Advances in Engineering and Management (IJAEM)
Volume 6, Issue 11 Nov. 2024, pp: 37-45 www.ijaem.net ISSN: 2395-5252

224×224×3. This layer applies 128 convolutional filters, each with a kernel size of 3 X 3, to the input images. The convolution operation can be mathematically expressed as:

(1) $Z^{(l)}=W^{(l)}*A^{(l-1)}+b^{(l)}$

Here in (1), $Z^{(l)}$ represents the output of the convolutional layer, $W^{(l)}$ denotes the filter weights, $A^{(l-1)}$ is the input image or the output from the previous layer, and $b^{(l)}$ is the bias term. The activation function applied to this layer is ReLU (Rectified Linear Unit), which is defined as:

$\text{ReLU}(x)=\max(0, x)$

2nd Convolutional Layer: The output from the 1st convolutional layer is passed to the 2nd convolutional layer, which utilizes 256 filters with the same 3×3 kernel size. The convolution operation for this layer is same as (1):

$Z^{(l)}=W^{(l)}*A^{(l-1)}+b^{(l)}$

followed by ReLU activation.

3rd Convolutional Layer: This layer has 256 filters with the same 3×3 kernel size. The convolution operation for this layer is same as (1):

$Z^{(l)}=W^{(l)}*A^{(l-1)}+b^{(l)}$

followed by ReLU activation.

4th Convolutional Layer: The final convolutional layer also utilizes 256 filters with the same 3×3 kernel size. The convolution operation for this layer is same as (1):

$Z^{(l)}=W^{(l)}*A^{(l-1)}+b^{(l)}$

followed by ReLU activation.

Max Pooling Layers

Max pooling is applied after each convolutional layer to down-sample the feature maps and reduce spatial dimensions. The pooling operation uses a 2×2 pool size and selects the maximum value within each 2×2 region of the feature map. The formula for max pooling is:

(2) $A_{pooled}^{(l)} = \max_{f_0}(A_{region}^{(l)})$

where $A_{region}^{(l)}$ represents the values within the pooling region.

Dropout Layers

Dropout is employed to prevent overfitting by randomly setting a fraction of input units to zero during training. For convolutional layers, the dropout rate is set to 0.4, and for fully connected layers, it ranges between 0.2 and 0.4. The dropout operation is defined as:

Dropout(x)={ x with probability (1−p), 0 with probability p

where p is the dropout rate.

Fully Connected Layers

1st Dense Layer: The output from the last convolutional layer, after flattening, is fed into a fully connected layer with 512 neurons. Each neuron computes a weighted sum of inputs followed by ReLU activation:

$Z^{(l)}=W^{(l)}*A^{(l-1)}+b^{(l)}$

(3) $\text{ReLU}(Z^{(l)}) = \max(0, Z^{(l)})$

2nd Dense Layer: This layer has 64 neurons, and similarly computes:

$Z^{(l)}=W^{(l)}*A^{(l-1)}+b^{(l)}$

followed by ReLU activation.

3rd Dense Layer: This layer has 256 neurons, and similarly computes:

$Z^{(l)}=W^{(l)}*A^{(l-1)}+b^{(l)}$

followed by ReLU activation.

4th and 5th Dense Layers: These layers have 64 and 256 neurons, respectively. The operations are:

$Z^{(l)}=W^{(l)}*A^{(l-1)}+b^{(l)}$

followed by ReLU activation.

Output Layer

The output layer is a dense layer with 36 neurons, corresponding to the 36 classes (A-Z, 0-9). The softmax activation function is applied to normalize the output into a probability distribution:

(4) $\text{Softmax}(Z^{(l)}) = e^{Z(l)} / \sum_j e^{zj(l)}$

Where $e^{zj(l)}$ is the exponentiated output of the j-th neuron, and the denominator is the sum of exponentiated outputs over all neurons.

The CNN model was compiled using the Adam optimizer, which is well-suited for efficient gradient-based optimization. Categorical cross-entropy was chosen as the loss function since the task involved multi-class classification, with 36 different classes representing ASL gestures. The training process was carried out over 100 epochs using a batch size determined by the capacity of the training hardware. The dataset was fed to the model through train_generator, which supplied the preprocessed images in batches. For each epoch, the model performed a forward pass, computed the loss using the cross-entropy formula, and backpropagated the error to update the weights using Adam's optimization strategy. The training progress was monitored using validation data provided by validation_generator, which was

International Journal of Advances in Engineering and Management (IJAEM)
Volume 6, Issue 11 Nov. 2024, pp: 37-45 www.ijaem.net ISSN: 2395-5252

essential for preventing overfitting and ensuring generalization to unseen data.The final trained model was evaluated on a test dataset, measuring its accuracy across different classes.

D. Gesture Classification

The trained CNN model was loaded using a custom Python script that enables real-time gesture classification. The model configuration was stored in JSON format, and the weights were loaded from an HDF5 file. The script utilizes OpenCV to capture live video feed from the webcam. The region of interest (ROI) in the captured frame is cropped, converted to grayscale, and resized to 48x48 pixels, matching the input size of the model.

E. Gesture Prediction

The real-time prediction of gestures is performed using the loaded CNN model. The model predicts the gesture shown in the live video frame and displays the predicted label. Continuous predictions are made over time, and the script shows the recognized gesture on the screen.

IV. SYSTEM ARCHITECTURE

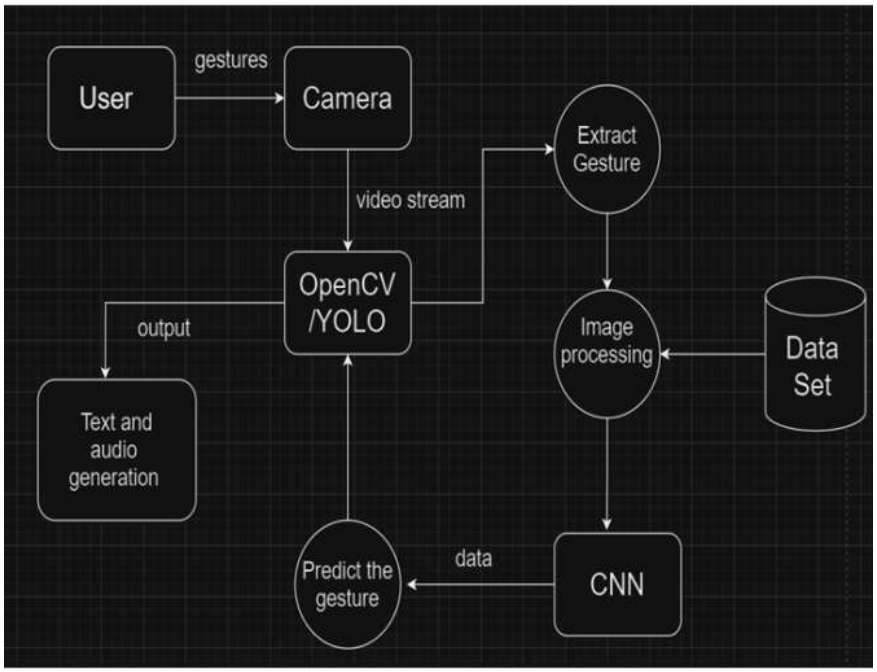

Fig. 1. System Architecture

A. User Provides Gestures

The user performs sign language gestures in front of a camera.These gestures are captured as video frames by the camera.

B. Capture Video Frames

Utilize OpenCV (Open Source Computer Vision Library) to access the video stream from the camera.Continuously capture frames from the video stream.

C. Gesture Extraction

For each captured frame, isolate the region of interest (ROI) containing the hand or gesture.Apply techniques like background subtraction or color thresholding to separate the hand from the background.

D. Image Processing

Preprocess the extracted gesture image to enhance its features and reduce noise.Resize the image to a standard size suitable for processing.Normalize pixel values to ensure consistent input for the machine learning model.Optionally, perform edge detection or other image enhancements to emphasize gesture contours.

E. Load Dataset

Load a pre-existing dataset of sign language gestures.Each gesture in the dataset is associated with a label that denotes the sign it represents.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 6, Issue 11 Nov. 2024, pp: 37-45 www.ijaem.net ISSN: 2395-5252

F. Comparison Using CNN (Convolutional Neural Network)

Employ a CNN model that has been trained on the sign language dataset.Feed the preprocessed image (from step 4) into the CNN model.Compare features extracted from the processed image with features learned from images in the dataset.The CNN computes similarities and identifies the closest match between the user's gesture and gestures in the dataset.

G. Gesture Prediction

Based on the comparison, the CNN predicts which sign gesture the user is making.The prediction corresponds to the label associated with the most similar gesture from the dataset.

H. Display Predicted Sign:

Use OpenCV to display the predicted sign (label) on the screen in a clear and understandable format.Overlay text or an image representing the predicted sign gesture for user feedback.

I. Convert Prediction to Audio:

Convert the predicted sign (label) into audio output for accessibility or additional feedback.Implement text-to-speech (TTS) functionality using libraries or services to convert the predicted label into spoken language.

V. DATASET

Fig.2. Dataset

Fig.3.Dataset of ASL alphabet and numbers(0-9):The figure presents a comprehsive dataset of the American Sign Language (ASL) alphabet, including all 26 letters and the numbers 0-9. Each sign is represented visually, showcasing the specific hand shapes and movements used in ASL to communicate each letter and number.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 6, Issue 11 Nov. 2024, pp: 37-45 www.ijaem.net ISSN: 2395-5252

VI. RESULTS

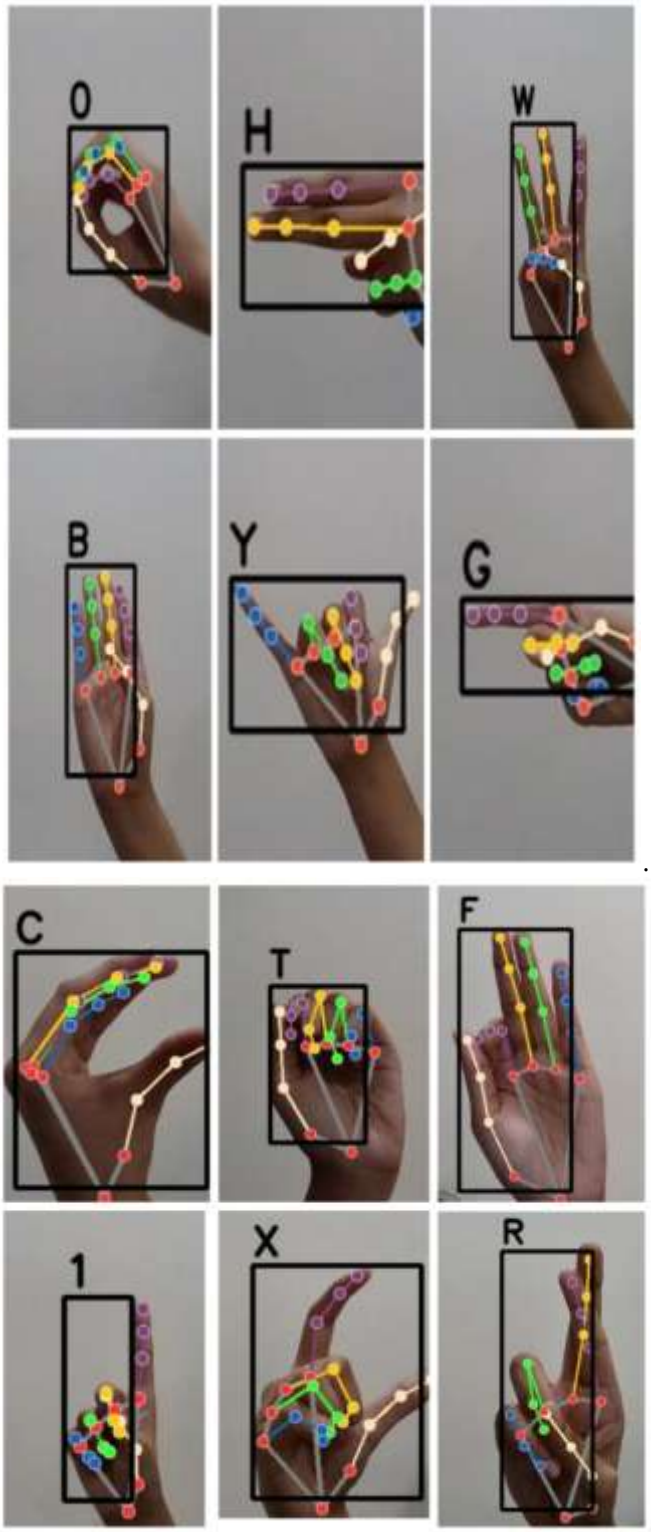

Fig.4. Real Time Sign Detection

International Journal of Advances in Engineering and Management (IJAEM)
Volume 6, Issue 11 Nov. 2024, pp: 37-45 www.ijaem.net ISSN: 2395-5252

Table 1:Accuracies of each class : The Accuracy of the entire model is approximately 85.87%.

Class	Accuracy (in %)
0	75
1	92.31
2	83.33
3	92.86
4	81.21
5	89.23
6	81.82
7	93.44
8	87.77
9	93.24
A	85.45
B	100
C	95
D	84.56
E	89.89
F	93
G	100
H	100
I	88.72
J	94.43
K	92.25
L	96
M	66.67
N	82.98
O	83.33
P	87.67
Q	79
R	90.91
S	66.67
T	50
U	90.89
V	84.62
W	91.67
X	92.23
Y	100
Z	78.34

VII. CONCLUSION

In summary, the creation and implementation of a sign language detection system using convolutional neural networks (CNNs) mark a notable advancement in applying machine learning to improve accessibility and communication technology. This project sought to address the pressing need for effective communication tools for individuals who are deaf or hard of hearing, enhancing their ability to interact within educational, professional, and social settings.Throughout the project, we examined the methodologies and technical complexities of training CNN models to accurately recognize and interpret sign language gestures. By utilizing CNNs, known for their strength in capturing spatial relationships and hierarchical features from visual data, our aim was to achieve a high level of accuracy in recognizing both static and dynamic gestures in real-time. The focus on multi-language support highlights our commitment to inclusivity, ensuring the system can accommodate a variety of sign languages from around the world.The system's functional requirements emphasized its ability to operate in real-time, adapt to changing environmental conditions, and integrate seamlessly with existing technologies. A key aspect of ensuring its reliability was the system's resilience to noise, varying lighting, and different hand orientations across diverse user scenarios.Looking ahead, continued advancements in deep learning and the expansion of sign language datasets will further improve the accuracy and capabilities of such systems. Future research could explore hybrid models that combine CNNs with other machine learning techniques or incorporate multi-modal inputs for more comprehensive gesture recognition.In conclusion, this project contributes to the broader mission of promoting inclusivity and accessibility through technology, empowering those who use sign language as their primary mode of communication. By breaking down communication barriers, we aim to create more equitable opportunities and improve the quality of interactions for all individuals, regardless of their communication needs.

REFERENCES

[1] K. Bantupalli and Y. Xie, "American sign language recognition using machine learning and computer vision," DigitalCommons@Kennesaw State University, Feb. 18, 2019.

[2] N. Ahmad, E. S. Wijaya, and I. A. Iswanto, "Transforming Sign Language using CNN Approach based on BISINDO Dataset," in Proceedings of the 2023 International Conference on Informatics, Multimedia, Cyber and Information Systems (ICIMCIS), 2023.

[3] V. A. Akano and A. O. Olamiti, "Conversion of sign language to text and speech using machine learning techniques," Journal of Research and Review in Science, vol. 5, no. 1, pp. 58-65, Dec. 2018.

[4] S. Lang, M. Block-Berlitz, and R. Rojas, "Sign language recognition using Kinect," in Proceedings of the 11th International Conference on Artificial Intelligence and Soft Computing - Volume Part I, Springer, Apr. 2012.

[5] Ben Haj Amor, O. El Ghoul, and M. Jemni, "Sign Language Recognition Using the Electromyographic Signal: A Systematic Literature Review," Sensors, vol. 23, no. 19, pp. 8343, Oct. 2023

[6] Katoch, S., Singh, V., & Tiwary, U. S. (2022). Indian sign language recognition system using SURF with SVM and CNN. Version of Record 27 April 2022.

[7] R. De Souza and E. B. Pizzolato, "Sign language recognition with support vector machines and hidden conditional random fields: Going from fingerspelling to natural articulated words," in Proceedings of the 9th International Conference on Machine Learning and Data Mining in Pattern Recognition, vol. 7988, pp. 84-98, Jul. 2013.

[8] S. Agre, S. Wasker, A. Vashishtha, H. Latkar, and A. Kanse, "Real-time conversion of sign language to text and speech, and vice-versa," Journal of Emerging Technologies and Innovative Research (JETIR), vol. 10, no. 11, Nov. 2023.

[9] R. M. Kagalkar and S. Gumaste, "Mobile application-based translation of sign language to text description in Kannada language," International Journal of Interactive Mobile Technologies (iJIM), vol. 12, no. 2, pp. 92, Mar. 2018.

[10] K. M. Tripathi, P. Kamat, S. Patil, R. Jayaswal, S. Ahirrao, and K. Kotecha, "Gesture-to-text translation using SURF for Indian sign language," Applied System Innovation, vol. 6, no. 2, pp. 35, 2023.

[11] K. Manikandan, A. Patidar, P. Walia, and A. B. Roy, "Hand gesture detection and conversion to speech and text", 2018.

[12] M. Prabhakar, P. Hundekar, S. B. Deepthi, and S. Tiwari, "Sign language conversion to text and speech," Journal of Emerging

Technologies and Innovative Research (JETIR), vol. 9, no. 7, Jul. 2022.
[13] M. Kumar, "Conversion of sign language into text," ,2018
[14] R. Harini, R. Janani, S. Keerthana, S. Madhubala, and S. Venkatasubramanian, "Sign language translation," in Proceedings of the 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Mar. 2020.
[15] T. Ananthanarayana, P. Srivastava, A. Chintha, A. Santha, B. Landy, J. Panaro, A. Webster, N. Kotecha, S. Sah, T. Sarchet, and R. Ptucha, "Deep learning methods for sign language translation," ACM Transactions on Accessible Computing (TACCESS), vol. 14, no. 4, Article 22, pp. 1-30, Oct. 2021.
[16] Z. Liang, H. Li, and J. Chai, "Sign language translation: A survey of approaches and techniques," Electronics, vol. 12, no. 12, pp. 2678, Jun. 2023.
[17] Bragg, O. Koller, M. Bellard, L. Berke, P. Boudreault, A. Braffort, N. Caselli, M. Huenerfauth, H. Kacorri, T. Verhoef, C. Vogler, and M. R. Morris, "Sign language recognition, generation, and translation: An interdisciplinary perspective," in Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '19), pp. 16-31, Oct. 2019.
[18] Guo, W. Zhou, H. Li, and M. Wang, “Hierarchical LSTM for Sign Language Translation”, AAAI, vol. 32, no. 1, Apr. 2018.
[19] L.-J. Kau, W.-L. Su, P.-J. Yu, and S.-J. We, "A real-time portable sign language translation system," in Proceedings of the 2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS), Aug. 2015
[20] “Sign Language Translation Using Deep Convolutional Neural Networks,” KSII Transactions on Internet and Information Systems, vol. 14, no. 2. Korean Society for Internet Information (KSII), 29-Feb-2020.