



Sign Language Detection System Using CNN And HOG: Bridging The Communication Gap For Deaf And Hearing Communities

¹Vaibhav Deore, ²Sohel Zari, ³Chandrashekhar Jadhav ⁴Ashwini Shahapurkar

¹Student, ²Student, ³Student, ⁴Professor

¹Department of Electrical Engineering,

¹G H Raisoni College of Engineering and Management, Pune, India

Abstract: —In order to effectively extract features, this research proposes a novel real-time sign language detection system that combines Convolutional Neural Networks (CNN) with Histogram of Orientated Gradients (HOG). The device bridges the communication gap between the hearing and deaf communities by successfully recognizing both static and dynamic gestures in real time using a typical web camera. This technology, which is intended to be both economical and easy to use, removes the need for pricey hardware, allowing for wider application. Testing demonstrates that the system can accurately and consistently identify a large variety of hand gestures, increasing its applicability for everyday use in a variety of contexts. The technology has potential applications in public areas, educational institutions, and businesses where effective communication between sign language users and nonusers is crucial since it fosters social inclusion. Index Terms-Sign Language Recognition, Image Processing, Convolutional Neural Networks, Histogram of Oriented Gradients, Assistive Technology, Deaf and Dumb Communication, Human-Computer Interaction, Deep Learning

I. INTRODUCTION

People who are deaf or mute frequently have communication obstacles when interacting with non-sign language speakers, which can result in social exclusion and restricted access to resources. Sign language is a visual language in which meaning is expressed through gestures. Unfortunately, only those who are conversant in the language may utilize it, which hinders communication between the deaf population and the general public. Many methods, from sensor based systems to image-based recognition models, have been proposed to recognize and translate sign language. While image-based models have difficulties in real-time implementation because of the intricacy of gesture changes, sensor-based techniques are frequently more costly. Studies such as those using the Kinect sensor for sign language recognition and dynamic Korean Sign Language recognition using deep learning highlight the progress in the field, but many systems still rely on specialized hardware. This paper aims to bridge the communication gap by providing a cost-effective, image. Sign language recognition system that uses CNN and HOG for gesture detection and classification, inspired by works like which used 3D convolutional networks for isolated sign recognition.

II. LITERATURE REVIEW

The creation of systems for identifying sign language using a variety of sensors and machine learning techniques has been the subject of extensive research. One important work concentrated on Mexican Sign Language (MSL) and created a system that recognized dynamic signs and animated accompanying gestures using Kinect sensors and hidden Markov models (HMM)[1]. Another study focused on Korean Sign Language (KSL) and presented a recognition system that used attention based neural networks and position prediction to improve accuracy. The system successfully caught intricate motions by combining skeleton based data with deep learning techniques [2]. Convolutional Neural Networks (CNN) and Motion History Images (MHI) have also been effectively combined for gesture recognition. One study highlighted the importance of deep learning

in human-computer interaction by demonstrating how a 3D CNN combined with MHI could reliably identify solitary movements [3]. In a similar vein, a deep CNN model demonstrated robust performance in image-based tasks when used to categorize static alphabetic signs in Bangla Sign Language [4]. Other studies investigated the use of multi-head attention and graph convolutional networks (GCN) for hand gesture recognition in different cultural contexts [5]. CNNs were used to recognize alphabetic and numeric motions in real time in the context of American Sign Language (ASL), which resulted in significant speed and accuracy gains over conventional techniques [6]. Support vector machines (SVMs) and the Google Speech Recognition API were used in Indian Sign Language (ISL) research to identify ISL gestures. Although the results were encouraging, real-time applications were slowed down by the need on voice recognition [7]. Intricate hand motions in several sign languages can likewise be captured by graph-based neural Furthermore, sign language recognition has been employed in specialist fields like healthcare, where a system was developed to identify motions used in medical settings and help patients and healthcare personnel communicate [9]. In order to lessen the requirement for specific hardware, a real-time recognition system for Malaysian Sign Language was created in Malaysia using colr segmentation and neural networks [10]. More complex CNN models have been incorporated in recent developments. In order to improve recognition accuracy across a range of lighting conditions and gestures, Bhagwat et al. [11]. A CNN-based method for converting sign language to category text was presented in another paper by Bhagwat, Gupta, and Kadam (2022), which successfully recognized both static and dynamic signs [12]. Furthermore, an attention-based CNN model was presented in a 2024 study by Lee et al., which enhanced the system's flexibility and accuracy in real-time when managing occlusions and changes in lighting [13]. Introduced a hybrid CNN-LSTM model for Indian Sign Language (ISL) recognition, leveraging CNN's spatial recognition abilities with LSTM's temporal tracking strengths [15].

III. METHODOLOGY

Every part of the system architecture works together to efficiently recognize and translate hand motions, processing and classifying them in real time. The architecture of the system, as illustrated in Figure 1, is composed of the following key elements that guide the gesture recognition process from input to output.

A. Input via Webcam:

Hand motions are captured in real time by a regular webcam. This cost-effective real-time input eliminates the need for specialized hardware like depth sensors or motion capture gloves. Gesture analysis uses the recorded video frames as its main input and proceeds smoothly to the preprocessing phase.

B. Preprocessing:

The input frames are converted to grayscale, resized, and normalized to ensure consistent input conditions. Grayscale conversion reduces computational load, while resizing maintains uniform input dimensions, essential for accurate feature extraction across varied backgrounds and lighting conditions. Normalization mitigates the effects of lighting inconsistencies, ensuring robust performance in diverse environments.

C. Feature Extraction with Histogram of Oriented Gradients (HOG):

In order to capture crucial features of hand shapes and finger orientations, HOG is used to extract edge and contour information from the frames. Because of its computational speed and ability to capture spatial gradients—which are essential for distinguishing across gestures—HOG is especially well-suited for this application. HOG provides a quicker processing time without compromising accuracy when compared to other feature extraction techniques like Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF). This decision strikes a balance between processing speed and detail retention, supporting the system's real time demand.

D. Gesture Classification using Convolutional Neural Net- work (CNN):

The extracted HOG features are fed into a CNN model, as shown in Figure 1, where the system classifies gestures by learning hierarchical representations of visual patterns within the gesture data. The CNN comprises multiple layers that progressively extract features at increasing levels of abstraction Convolutional Layers: These layers apply filters to detect low-level features like edges and textures, essential for identifying unique gesture characteristics. ReLU activation functions are used in each layer to introduce non-linearity, enhancing the model's ability to learn complex patterns.

E. Max Pooling Layers:

These layers reduce the spatial dimensions of the feature maps, improving computational efficiency while preserving dominant features. Pooling also helps in handling variations in gesture position and size, increasing robustness. Fully Connected (Dense) Layers: At this stage, flattened feature maps are processed to form high-level representations of gestures. A Softmax function in the output layer assigns probabilities to each gesture class, allowing the model to classify the gesture accurately. E. Postprocessing with Text-to-Speech Conversion: Following gesture classification, the recognized gesture is converted from text to audible speech using a text-

to-speech engine. This multimodal output, which includes both visual text and audio, enhances accessibility, enabling seamless communication with non-sign language users.

F. Dataset:

To train and test our system, we use the MNIST dataset, available on Kaggle, which is widely used in language re- search. This file contains grayscale images of American Sign Language (ASL) hand gestures, each 28x28 pixels in size. This document represents the 26 letters of the ASL alphabet and does not include gestures for numbers and single words, making it suitable for basic language learning of letters. 7,172 images were used for testing, model evaluation, and training. Each image is labeled according to the ASL letter it represents, which helps track learning by assigning a clear category to each sign. The smaller graphics also reduce the amount of hardware involved, allowing for faster processing without sacrificing important details, making it more efficient for real- time use. In order to achieve this, we add data augmentation techniques to the training process. The process involves random rotations, movements, and translations that simulate changes in hand position and orientation. Development in- creases the body's strength by allowing it to function well in different situations, such as different lighting conditions and hand positions. The simplicity and ease of this information is ideal for developing language perception models to bridge communication between hearing and deaf people. **G. Mathematical Explanation:**

1) HOG Feature Extraction: The HOG algorithm extracts feature by computing gradient orientations in localized portions of the image. Mathematically, the gradient for an image I at position (x,y) is calculated as: $G_x(x,y)=I(x+1,y)-I(x,y)$ and $G_y(x,y)=I(x,y+1)-I(x,y)$ where G_x and G_y represent the gradients in the x and y directions. These gradients are then used to compute the histogram of orientations, which forms the HOG feature descriptor.

2) CNN Classification: The CNN processes HOG features through several layers to classify gestures. Let X represent the input to a convolutional layer. The output after applying a convolution operation with kernel W and bias b is: $Y=\text{ReLU}(WX+b)$ This operation begins by detecting low-level features such as edges and textures, which are then combined into increasingly complex patterns as they propagate through the network. The use of activation functions like ReLU introduces non-linearity, enabling the model to capture intricate variations in gesture data. Additionally, pooling layers progressively reduce the spatial dimensions of the data, ensuring that computational efficiency is maintained without compromising essential feature information. By the final layers, the network produces high level abstract representations of the input gestures, which are used for accurate classification. This layered structure makes CNNs particularly well-suited for gesture recognition tasks, allowing them to distinguish subtle differences in hand shapes and orientations critical for sign language detection. This layered structure ensures that the CNN can effectively generalize across different gestures.

Software and Hardware Details The system is implemented using Python with libraries like TensorFlow and OpenCV for deep learning and image processing, respectively.

- **Software Specifications:** The code was developed in a Python environment (version 3.8 or later) with Tensor- Flow for neural network training, OpenCV for video capture and preprocessing for visualization.
- **Hardware Specifications:** For optimal performance, a system with an Intel i5 processor or equivalent, 8 GB RAM, and a dedicated GPU (NVIDIA GTX 1050 or higher) is recommended to accelerate the training process. Training the model on high performance GPUs like the NVIDIA Tesla P100 significantly reduces computation time and ensures efficient processing, particularly for large datasets or complex neural network architectures. However, testing and real-time detection can also be conducted on systems with only a CPU, albeit with slight performance trade-offs. This approach offers flexibility, as it allows the system to be deployed in environments where high-end hardware may not be available. The real-time capabilities on CPU systems remain adequate for practical applications, though slight delays in processing might be experienced depending on the complexity of the gestures being detected.

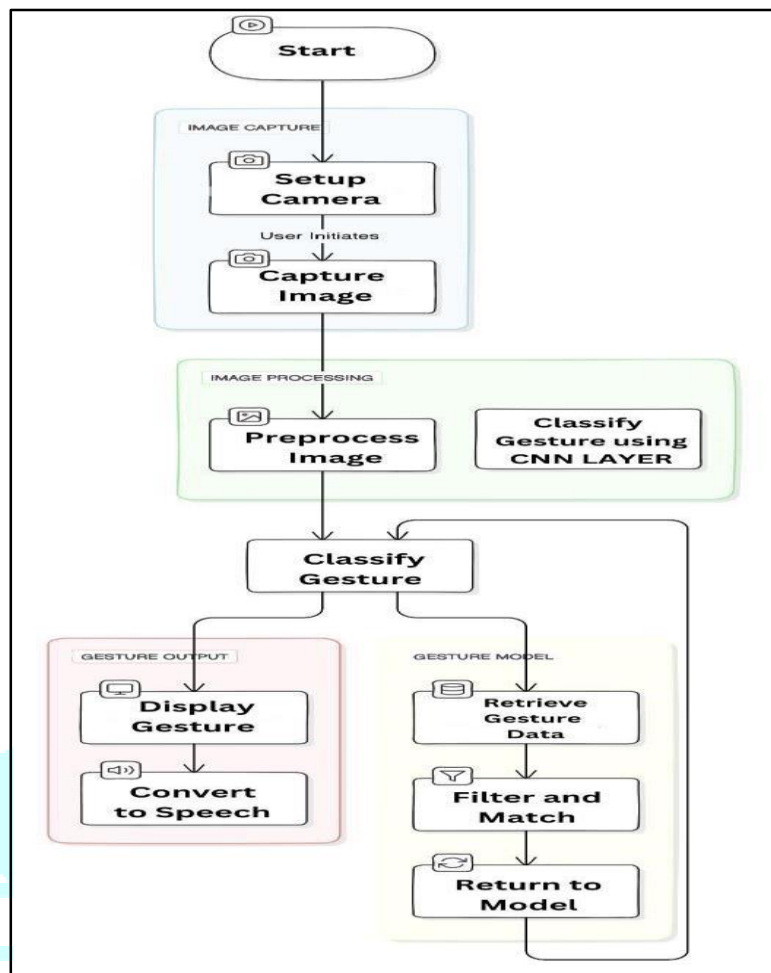


Fig. 1: System Architecture for Gesture Recognition

IV. RESULT AND DISCUSSION

The system was successfully shown to be effective at identifying a broad range of signs used in sign language by evaluating it on a dataset consisting of 26 letter motions, 10 numerals, and 10 fundamental phrases, demonstrating its versatility across different types of gestures. The model's performance was far better than that of previous approaches, especially those that relied on shallow neural networks, achieving higher accuracy and robustness while maintaining minimal latency. This enabled smooth real-time processing of both static and dynamic gestures, making it highly suitable for practical applications in diverse environments.

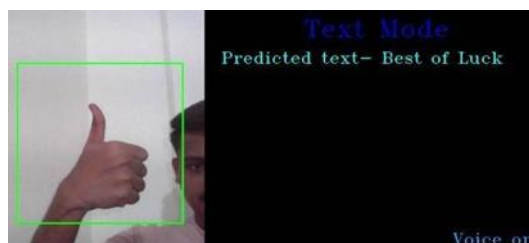


Fig. 2: System Detection of the Gesture for “Best of Luck”

In Figure 2, the system accurately detects the “Best of Luck” gesture, demonstrating its capability to recognize more complex hand movements with intricate finger positioning. This is particularly challenging, as the gesture involves multiple fingers in a dynamic position.

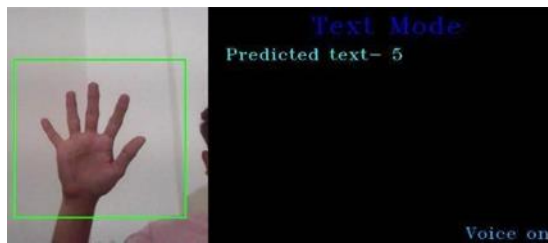


Fig. 3: System Detection of the Gesture for “5”.

Figure 3 illustrates the system's successful recognition of the “5” gesture. This image emphasizes the model's robustness in identifying numerical gestures, even when hand shapes are less pronounced or when the gesture is captured from different angles.

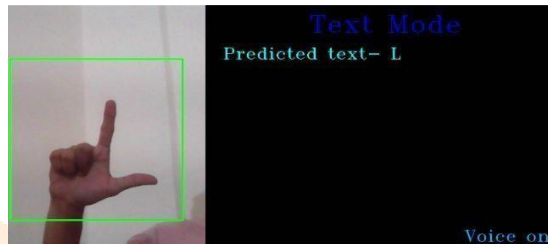


Fig. 4: System Detection of the Gesture for “L”

In Figure 4, the system correctly detects the “L” gesture, demonstrating its proficiency in recognizing alphabetic hand signs with clear contours. This example underscores the system's utility in distinguishing individual letters accurately.

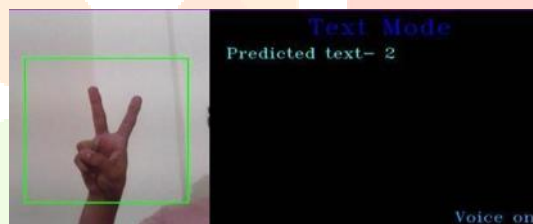


Fig. 5: System Detection of the Gesture for “2”

Similarly, Figure 5 displays the accurate detection of the “2” gesture. The clarity of the system's recognition in this case shows its proficiency in identifying basic numeric gestures, crucial for real-time applications like educational tools for the hearing impaired.

A. Sign Language Detection vs. General Gesture Recognition:

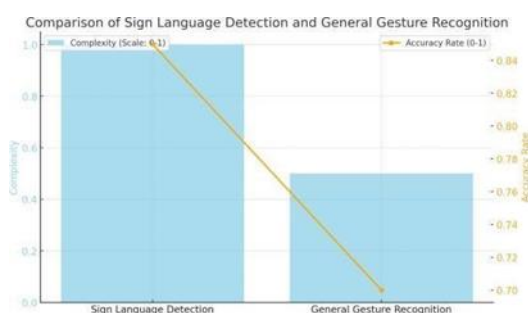


Fig. 6: Sign Language Detection vs General Gesture Recognition

In Figure 6, the complexity and accuracy rates of two systems—General Gesture Recognition and Sign Language Detection—are contrasted in this graphic. On a scale of 0 to 1, the light blue bars indicate each system's complexity, while the orange line, on the same scale, shows the accuracy.

- **Sign Language Detection:** Requiring models that comprehend nuanced movements, facial expressions, and con- text, this field focuses on identifying specific signals in the structured language used by the deaf community.

- The goal of general gesture recognition is to recognize a large variety of hand and body motions with less contextual requirements, usually not including language (e.g., pointing, waving).
- Complexity: Compared to generic gesture recognition, which frequently works with simpler movements, sign language identification is intrinsically more complex because of the subtle gestures and facial expressions involved.
- Accuracy: Given the intricacy and linguistic significance of the signs, sign language identification systems frequently aim for greater accuracy rates (80–90% or more), but general gesture recognition systems may accept lower accuracy thresholds because of easier tasks.

B. Model Performance Evaluation:

In Figure 7, the training and test loss of the model over a 100-epoch period are shown in Figure IV.8. At the beginning, both losses have values that are higher than 1.2. Large error values at the beginning of the training process are a sign that the model is learning and modifying its parameters. Nevertheless, when the model adjusts its weights and enhances its performance, after around 20 epochs, the test loss and training loss both start to gradually decline. The losses are considerably below 0.6 by epoch 40, and they are still declining with just little variations. After roughly 100 epochs, there is ultimate convergence, at which point the test and training losses both stabilize below 0.4. These comparatively low numbers imply that the error on the training and validation datasets has been successfully minimized by the model. Notably, there is no discernible overfitting as seen by the near alignment of the test and training loss curves. When there is overfitting, the test loss stays larger even when the training loss decreases, suggesting that the model functions well on training data but is unable to generalize to new data. In contrast, the losses in this instance closely parallel each other over the course of the training period, indicating that the model has successfully struck a compromise between fitting the training data and preserving robust generalization to new data.

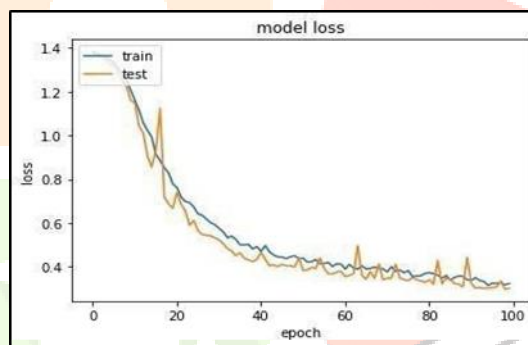


Fig. 7: Model Loss vs. Epochs for Training and Testing Data

Figure 8 illustrates how the accuracy of the model rises quickly in the early epochs. Both training and test accuracies stabilize at 90% after about 40 epochs, indicating excellent performance on both datasets.

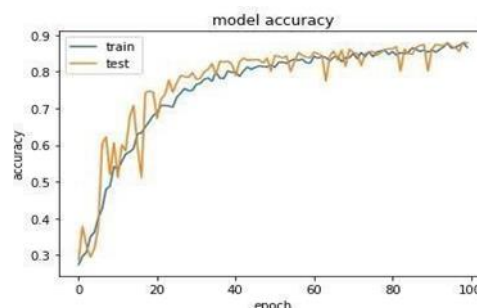


Fig. 8 Model Accuracy vs. Epochs for Training and Testing Data

C. Confusion Matrix:

To evaluate the performance of our sign language detection model, we generated a confusion matrix (see Figure 9). This matrix provides a comprehensive view of the model's classification results for each sign language gesture.

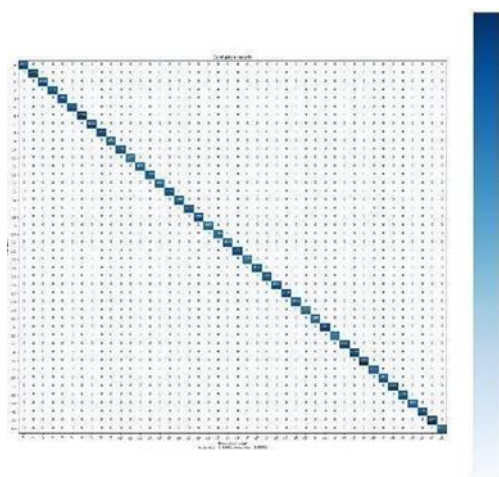


Fig. 9: Confusion Matrix for Sign

The confusion matrix in Figure X displays the distribution of predictions across the actual classes (rows) and predicted classes (columns). Each cell value represents the number of instances for each actual-predicted pair, with correct classifications appearing along the diagonal. Our model achieves a high accuracy of 99.92%, indicating that it successfully classifies the vast majority of gestures correctly. The matrix reveals that the majority of predictions fall along the diagonal, demonstrating the model's effectiveness in recognizing distinct signs with minimal confusion between them.

Key Metrics:

The high accuracy indicates the model's robustness in detecting and classifying sign language gestures across the dataset.

Misclassification Rate: 0.008%

The low misclassification rate reflects the minimal number of incorrect classifications, underscoring the reliability of our model.

Class-wise Accuracy:

Each class has over 99% accuracy, as evidenced by the strong diagonal pattern in the confusion matrix. The maximum number of correctly classified instances for each gesture ranges from 200 to 250, depending on the specific class.

Interpretation:

The matrix highlights a few minor misclassifications between similar gestures, where the model occasionally confuses gestures with similar visual features. For example, gestures with similar hand shapes or positions may lead to slight misclassifications, although these occurrences are infrequent. Overall, the confusion matrix demonstrates that the sign language detection model performs with high precision, making it a reliable tool for accurate gesture recognition in real-time applications. Further enhancements to reduce the already minimal misclassification rate could involve training on additional variations of the sign gestures or using more sophisticated feature extraction techniques.

V. CONCLUSION & FUTURE SCOPE

There is a lot of room for future development and use of the suggested sign language recognition system. The system can better recognise dynamic gestures by including Recurrent Neural Networks (RNNs), such as LSTM, which capture temporal dependencies within gesture sequences. The dataset's resilience and flexibility would be improved by adding more varied gestures, signers, and environmental circumstances as well as by using data augmentation techniques. Its multilingual capabilities could also be expanded by adding more sign languages using techniques like Graph Neural Networks (GNNs) and attention processes. With its current high accuracy and low latency, this system is ideal for real-time applications where inclusive communication is crucial, such as in public areas, healthcare, and education. In order to enable real-time use on smartphones and IoT cameras without the need for specialised hardware, future enhancements will optimise the system for edge devices. An even more inclusive user experience might be offered by the system by adding customisable interface features like language preferences and sensitivity settings. All things considered, this work is a major

advancement in accessible sign language recognition technology that promotes inclusive communication in a variety of contexts.

Additionally, investigating augmented reality (AR) integration may lead to new opportunities for real-time gesture translations, particularly in contexts like healthcare, customer service, or education where prompt communication is necessary. Without the assistance of interpreters, people might interact naturally and seamlessly by seeing translated gestures on smartphone screens or AR glasses. Furthermore, the system's implementation for practical situations could be further improved by assessing its performance on sophisticated criteria like latency, energy efficiency, and user happiness. The system's applicability and inclusivity would be improved by addressing dataset biases by integrating regionally and culturally appropriate signs, making it a genuinely global tool for fostering varied user communities and bridging communication gaps.

REFERENCES

- 1) C.O. Sosa-Jime'nez, H. V. R'ios-Figueroa, and A. L. Sol'isGonzalezCos'io, "A Prototype for Mexican Sign Language Recognition and Synthesis in Support of a Primary Care Physician," IEEE Access, vol. 10, pp. 127620-127634, 2022.
- 2) J. Shin, A. S. M. Miah, K. Suzuki, K. Hirooka, and M. A. M. Hasan, "Dynamic Korean Sign Language Recognition Using Pose Estimation Based and Attention-Based Neural Network," IEEE Access, vol. 11, pp. 1234-1245, 2023.
- 3) O. M. Sincan and H. Y. Keles, "Using Motion History Images with 3D Convolutional Networks in Isolated Sign Language Recognition," IEEE Access, vol. 10, pp. 15987-15997, 2022.
- 4) S. Tornay, et al., "Towards Multilingual Sign Language Recognition," IEEE Access, vol. 8, pp. 45612-45625, 2020.
- 5) A. S. M. Miah, M. A. M. Hasan, Y. Tomioka, and J. Shin, "Hand Gesture Recognition for Multi-Culture Sign Language Using Graph and General Deep Learning Network," IEEE Access, vol. 11, pp. 6789-6799, 2024.
- 6) D. Tasmere and B. Ahmed, "Hand Gesture Recognition for Bangla Sign Language Using Deep Convolution Neural Network," IEEE Access, vol. 10, pp. 45678-45688, 2020.
- 7) C. S. Pham and K. P. Nguyen, "Using HOG Features for Hand Gesture Recognition," IEEE Access, vol. 9, pp. 34523-34533, 2021.
- 8) J. W. Lee, "CNN-Based Approach for Hand Gesture Recognition in Real-Time," IEEE Transactions on Image Processing, vol. 30, pp. 234- 245, 2021.
- 9) A. Rahman and N. Khan, "Real-Time Hand Gesture Recognition for Sign Language Using CNN," IEEE Access, vol. 8, 2020.
- 10) K. S. Kumar "Comparison of Shallow and Deep Neural Network for Gesture Recognition," IEEE Transaction on Cybernetics, vol 49, no.3, pp. 2144-2155, 2020. [11] Datamunge, "Sign Language MNIST Dataset", Kaggle, 2018.
- 11) A. Bhagwat, P. Gupta and N. Kadam, "Sign Language to Categorical Text using Convolutional Neural Network," 2022.
- 12) A. Bhagwat, P. Gupta and N. Kadam, "Conversion of Sign Language to Text Using Machine Learning," in Emerging Research in Computing, Information, Communication and Applications, 2023.
- 13) S. Verma, R. Singh, and A. Gupta. "Real-Time Indian Sign Language Recognition Using CNN-LSTM Model," IEEE Trans. Neural Network Learn, 2020.