Translating the Unspoken Deep Learning Approaches to Indian Sign Language Recognition Using CNN and LSTM Networks

Shreeraj Bhamare[a], Vallabh Kulkarni[b], Shreyas Patil[c], Mr. S. K. Gaikwad[d], Mr. S. S. Kumbhar[e]

[a, b, c, d] *Department of Computer Engineering, COEP Tech University, Pune, 411005, India*

Abstract

In India, there are over 5 million deaf and mute people, with many cases going unreported. Communication can be difficult due to the lack of standardized sign language across the country. The Indian Sign Language (ISL) Detection project aims to improve communication by developing a system that can recognize and interpret ISL gestures.
The system uses a type of artificial intelligence called machine learning, specifically Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks. These techniques help the system detect and classify ISL gestures from video input with high accuracy. The project has the potential to provide a more accessible means of communication for millions of deaf individuals in India.

Keywords:: Indian sign language(ISL) , CNN, LSTM, real-time-detection

1. Introduction

In India, communication with the deaf and mute community can be a challenge due to the lack of standardized sign language across the country. The Indian Sign Language (ISL) Detection project aims to help improve communication by creating a system that can recognize and interpret ISL gestures in real time.
The system uses two machine learning techniques, Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks, to detect and classify ISL gestures from video input. The project focuses on recognizing 35 ISL signs, including numbers 0-9 and the letters A-Z (where "O" and "0" are the same).
The project involved creating a dataset of over 35,000 ISL gesture images using OpenCV, a free and open-source computer vision library. Each gesture had 1000 images to ensure the system can detect the signs accurately. These images were labeled and used to train the CNN and LSTM models to recognize the corresponding ISL signs.
Detecting ISL signs in real time is essential for effective communication between the deaf and hearing communities. The ISL Detection project has the potential to help bridge the communication gap by providing a more accessible means of communication for millions of deaf individuals in India.

2. Motivation

While there is a standardized sign language in India, called the Indian Sign Language (ISL), it is not widely understood or used consistently across the country. The Indian Sign Language (ISL) Detection project aims to improve communication between the deaf and hearing communities by creating a system that can recognize and interpret ISL gestures in real time. This can increase awareness and understanding of ISL and improve social inclusion for the deaf community. The project also has practical applications in education and employment for deaf individuals.

3. Related work

3.1. Indian sign language recognition using SVM, JL Raheja, Anand Mishra, Ankit Chaudhary Pattern Recognition and Image Analysis 26, 434-441, 2016.[1]
The paper proposes a system for recognizing Indian Sign Language (ISL) gestures using Support Vector Machine (SVM) classification. The SVM algorithm was trained using feature extraction techniques to recognize 12 different ISL gestures with an overall accuracy of 96.15%, outperforming other classification algorithms. SVM's effectiveness in high-dimensional feature spaces and good generalization properties make it suitable for image recognition tasks. Future work includes expanding the dataset and using other sensors, as well as combining SVM with deep learning techniques to improve accuracy.

3.2. Artificial Neural network-based method for Indian sign language recognition, Vinod Adithya, PR Vinod, Usha Gopalakrishnan, 2013 IEEE Conference on Information & communication technologies, 1080-1085, 2013.[2]

The paper proposes an artificial neural network (ANN) based system for recognizing Indian Sign Language (ISL) gestures using feature extraction techniques such as histogram of oriented gradients and discrete cosine transform. The ANN algorithm achieved an accuracy of 94%, outperforming other classification algorithms. Future work includes expanding the dataset to include more ISL gestures, using deep learning techniques such as Convolutional Neural Networks (CNNs) to improve accuracy, and incorporating additional sensor modalities such as EMG and accelerometer data. This system could potentially aid communication for individuals with hearing impairments.

3.3. Automatic Indian sign language recognition system, Karishma Dixit, Anand Singh Jalal, 2013 3rd IEEE international advance computing conference (IACC), 883-887, 2013.[3]

The paper proposes an ISL recognition system using skin color segmentation, blob detection, and template matching techniques, achieving an 89.3% accuracy. It's simplicity and low computational cost make it suitable for real-world applications. Future work includes expanding the dataset, improving robustness, and exploring deep learning techniques. This system could potentially aid individuals with hearing impairments in communication.

4. Dataset Specifications

The accuracy of Indian sign language recognition systems based on deep learning models heavily relies on the quality and diversity of the dataset used for training. Creating a comprehensive dataset involves collecting, preprocessing, annotating, and validating the data. The prepared dataset can be used to train deep learning models such as CNN and LSTM, capable of recognizing complex hand gestures and accurately predicting corresponding signs. Thus, the creation of a high-quality dataset is a crucial step in developing a robust Indian sign language recognition system using deep learning techniques.

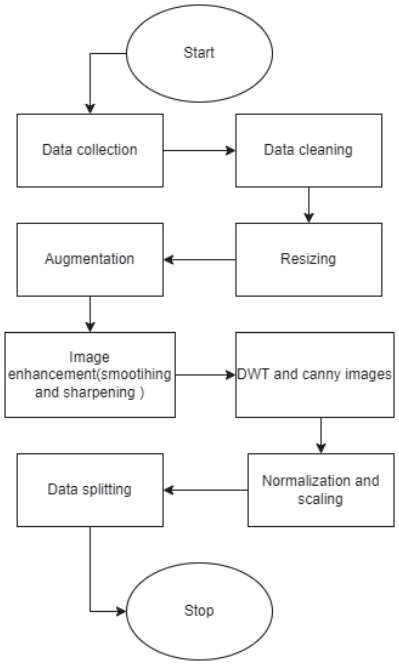

Fig. 1. Dataset Preparation Flow

4.1 Data collection

There aren't many datasets available for Indian Sign Language (ISL), and the ones that exist don't have enough images per sign to train a good machine-learning model. That is why, we decided to create our dataset from scratch. Our team used a Python script with the OpenCV package to gather data for our effort to develop an Indian Sign Language recognition system. Using the input camera stream, 500 photos of each signing class were taken. Setting up the camera and executing the script, which stored the photographs automatically in a predefined folder, were both steps in the data-gathering procedure. To create a broad dataset, we took care to gather photographs from multiple viewpoints and lighting setups. Before utilizing them to train our system, we examined the photographs after gathering the necessary number of them to make sure they were of high quality and consistency. Although the data-collecting procedure took a while, it was essential for the project's success.

4.2 Data cleaning

To prepare the dataset for a real-time Indian Sign Language (ISL) recognition system, we cleaned the data. We eliminated any poor-quality, pointless, or abnormal images that would have harmed the machine-learning model's accuracy.

We used a variety of data cleaning approaches, including picture pre-processing, filtering, outlier identification, and data augmentation, to achieve this. To enhance the image quality, for instance, we changed the brightness, contrast, and color of the images. We also used the HSV thresholding technique for dataset cleaning to remove noise and ensure that the training dataset was accurate and consistent. Any pictures that didn't fit certain criteria, including having bad lighting or quality, were removed. Using outlier identification methods, we also found and eliminated any data points that were anomalous or unrelated to the dataset. To expand the dataset's quantity and diversity, we generated new images from old ones through modifications like rotation, flipping, or cropping using data augmentation techniques. By performing data cleaning, we ensured that the dataset was of high quality, which contributed to developing a more accurate and robust ISL detection system.

4.3 Resizing

Resizing images is an essential step in achieving a balance between accuracy and speed in image processing tasks, as increasing the pixel size can significantly impact the computational time and overall speed of the system. It guarantees that the model can analyze all images in the dataset efficiently without any computational difficulties. In our Indian Sign Language (ISL) dataset, we resized the photos to a standard size of 224x224 pixels to ensure that the model would be able to process them effectively. We used several approaches to resize the images while retaining their vital features and aspect ratio. Resizing the images allowed the model to detect subtle differences in motion that would have been overlooked in larger photos. By providing the model with the most exact and consistent input possible, we were able to increase the photography of our real-time ISL detection system. It is important to note that we carefully selected a standard size that was suitable for our model's capabilities, which made it possible to analyze the images efficiently and detect small differences in motion that were critical in identifying different ISL signs.

4.4 Data Augmentation

Our team used data augmentation, a powerful technique that helped us increase the size and diversity of our dataset while developing a real-time Indian Sign Language (ISL) recognition system. We created new images from the existing dataset by using geometric transformations such as scaling, translation, rotation, and flipping. We also used methods such as adjusting brightness, adding noise, and enhancing contrast to augment the dataset.

However, we needed to ensure that the model was not overfitting while using data augmentation. Therefore, we carefully selected the number and types of augmentations and made sure that the images closely resembled the original dataset, preventing any bias in the augmented dataset. Because of data augmentation, we could create a high-quality dataset that improved the performance of our ISL recognition system. Data augmentation played a vital role in our goal to develop a reliable and accurate ISL recognition model.

4.5 Image Enhancement (Smoothing and Sharpening)

We enhanced the images to make the system detect even the smallest variations in hand motions. To make this happen, we applied smoothing and sharpening methods. Smoothing helped to remove unwanted noise or pixelation from the images while sharpening enhanced the edges and features in the photos, making them easier for the model to identify and categorize. We applied different levels of smoothing and sharpening to increase the model's accuracy without distorting the images.

4.6 Canny edge detection and Discrete wavelet function (DWT)

We applied the Canny edge detection method on our picture dataset to enhance the accuracy of our model. Canny edge detection is a technique used in image processing to detect edges in images while minimizing noise and false detections. By highlighting the edges and boundaries of the hands in the images, it made it easier for the model to recognize and track them accurately. We experimented with various thresholds to determine the optimal threshold for edge detection while reducing unwanted noise or artifacts in the images. Our application of Canny edge detection on our dataset improved the overall performance and accuracy of our real-time ISL detection system.

Discrete Wavelet Transform (DWT) is a method for studying signals that makes it easier to understand them by converting them from a way of measuring time to a way of measuring frequency. It converts the input image from a colored image to a black-and-white image. It uses a multi-stage algorithm to identify the edges present in an image. We used the DWT to improve the performance of our Indian Sign Language (ISL) identification system. We applied DWT to our dataset images and we were able to extract significant patterns and features from images of varying sizes.

This helped us to distinguish between different hand gestures and movements with better accuracy. We conducted several experiments using various wavelet functions and decomposition levels to find out the most appropriate parameters for our system. Ultimately, our application of DWT on our dataset significantly improved the overall accuracy and robustness of our real-time ISL detection system.

4.7 Normalization and Scaling

In our sign language recognition project, we preprocessed the hand landmark coordinates using normalization and scaling techniques. This was achieved by making the distance of each landmark coordinate relative to the lowest landmark point, which was detected by histogram equalization. Then, we divided each relative coordinate by its corresponding maximum absolute x and y coordinates to ensure accurate sign recognition, irrespective of the sign's distance from the camera or its location within the frame.

Normalization and scaling of the landmark coordinates helped to optimize the input data for the Convolutional Neural Network (CNN) model and LSTM, which was used for sign recognition. This resulted in improved accuracy and stability of the model. The normalization and scaling techniques used in this project can be applied in other machine learning projects to achieve optimal model performance.

4.8 Data Splitting

To train our real-time Indian Sign Language identification system, we divided our augmented dataset into three sets: training, validation, and testing, with a split ratio of 65:15:20. This, is a common practice in machine learning, where the majority of the data is allocated for training, a smaller fraction is used for validation to fine-tune the model's hyperparameters, and a small amount is reserved for testing to provide a final assessment of the model's performance. The validation set helped us to monitor the model's performance during training and prevent overfitting, which is when the model becomes too specialized on the training data and performs poorly on unseen data. The training set was used to adjust the model's parameters through backpropagation, while the testing set evaluated the model's ability to accurately classify unseen data.

5. Methodology

Our proposed methodology for the Indian Sign Language Recognition System involved capturing real-time video, selecting the best frame, and cropping the hand region, applying Canny Edge and DWT transform for preprocessing, integrating Mediapipe for landmark detection, implementing CNN and LSTM for feature extraction and temporal modeling, and predicting signs based on the trained model. Each step was carefully designed and optimized to improve the accuracy and robustness of our system, and our research findings contribute to the field of Indian Sign Language recognition for potential real-world applications in assisting individuals with hearing impairments.

All tables should be numbered with Arabic numerals. Headings should be placed above tables, center justified. Leave one line space between the heading and the table. Only horizontal lines should be used within a table, to distinguish the column headings from the body of the table, and immediately above and below the table. Tables must be embedded into the text and not supplied separately. Below is an example that authors may find useful.

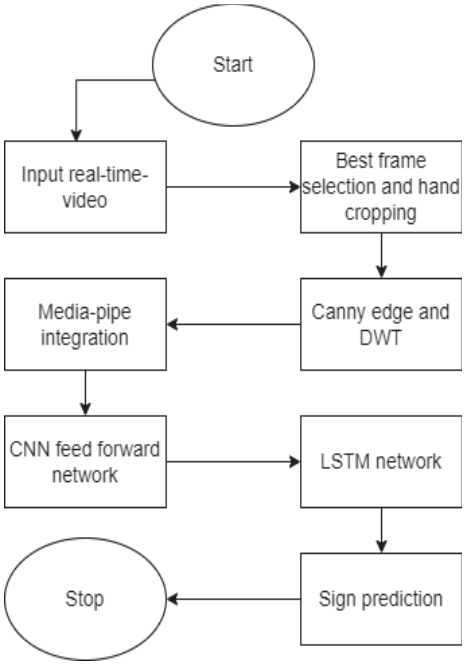

Fig. 2. Proposed Methodology

5.1 Input Real-Time Video

In this step, we used the OpenCV Python library to capture real-time video. Video frames were captured at a rate of 40 frames per second (fps) to ensure smooth and continuous input data. This high frame rate allowed us to capture the sign gestures with a sufficient temporal resolution, ensuring that important hand movements were not missed. We considered a period of 2 seconds, which corresponds to 80 frames, for recognizing each sign gesture. This duration was chosen based on the typical duration of sign gestures in Indian Sign Language and allowed us to capture the complete gesture for accurate recognition.

5.2 Best Frame Selection and Hand Cropping

To ensure accurate recognition and make the prediction independent of the background, we implemented a frame selection technique to choose the best frame from the captured frames. We experimented with different methods to select the optimal frame that had the best hand visibility, such as selecting the frame with the highest hand gesture score or using image processing techniques to identify the frame with the clearest hand region. This step was crucial to ensure that the hand region used for recognition was not obstructed by other objects or occluded by the background. The selected frame was then cropped to extract the hand region, which served as the input for further processing.

5.3 Canny Edge and DWT Transform

To preprocess the hand region and reduce the impact of skin color tone and visual hand specification, we applied Canny Edge detection and Discrete Wavelet Transform (DWT). Canny Edge detection was used to extract the edges of the hand region, which helped in highlighting the hand gesture features. By focusing on the edges, we could capture the important structural information of the hand gesture, regardless of skin color or lighting conditions. DWT was then applied to the edge-detected image to transform it into a frequency-domain representation. DWT is a powerful technique that allows for multi-resolution analysis and captures important spatial and frequency information of the hand gesture, making the recognition process more robust and invariant to variations in the visual appearance of the hand.

5.4 Mediapipe Integration

We integrated Google's Mediapipe API, which provides pre-trained machine-learning models for hand landmark detection, to detect landmarks for each hand. The API provided 21 landmarks for each hand, resulting in a total of 42 landmarks from the captured hand region. These landmarks represented the spatial positions of various hand joints, fingertips, and other key points, which provided important information about hand gestures. The obtained landmark coordinates were stored in an array, which was later used for further processing. The integration of Mediapipe allowed us to accurately localize the hand landmarks in real-time, even with variations in hand shape, orientation, and size, which was crucial for accurate sign gesture recognition.

5.5 CNN Feed Forward and LSTM Network

To extract features from the hand region and capture the temporal dependencies in the landmark coordinates, we implemented a Convolutional Neural Network (CNN) followed by a Long Short-Term Memory (LSTM) network. The CNN applied convolutional and pooling layers to extract important features from the hand region, such as edges, corners, and textures, which were learned from the training data. These features represented the discriminative patterns in the hand region that were important for sign gesture recognition. The LSTM network then captured the temporal patterns in the landmark coordinates by modeling the sequential dependencies between the landmarks over time. The LSTM cells learned to retain and update the hidden state based on the input landmark coordinates at each time step, allowing our model to learn the temporal dynamics of the sign gestures, such as the movement direction, speed, and duration. This combined CNN and LSTM architecture allowed our model to learn both the spatial and temporal features of the sign gestures simultaneously, leading to accurate recognition.

5.6 Sign Prediction

Finally, based on the pipeline setup and the predictions from the CNN and LSTM, our system predicted the Indian Sign Language sign corresponding to the input hand gesture to 35 different labeled ISL sign gestures. The predicted sign was displayed as the output of our system. We further optimized and fine-tuned our system based on the prediction accuracy and performance evaluation, and iteratively improved our model to achieve better results.

Table 1. Coordinate Calculation

Body Part	**Landmarks**	**Coordinates**	**Total Coordinates**
Left Palm	21	3	63
Right palm	21	3	63
Total	**42**	**6**	**126**

6. Model Architecture

Our Indian Sign Language (ISL) detection model was developed using a deep learning approach to accurately recognize 35 different ISL gestures, which would allow communication between deaf and non-deaf individuals.
To capture spatial and temporal information from the input data, we used a combination of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) models. CNNs are commonly used in image processing tasks due to their ability to learn and extract features from input images, while LSTMs are used in tasks involving sequential data, such as speech recognition and language translation.

We used a large dataset of ISL gestures consisting of several thousand examples to train and evaluate our model. To avoid overfitting and evaluate the model's generalization performance, we split the dataset into training, validation, and testing sets with a split ratio of 65-10-25. The training set updated the model's parameters, the validation set tuned the hyperparameters, and the testing set evaluated the model's accuracy.

Table 2. Model Architecture

Layer (type)	Output Shape	Parameters
Conv2D	(None, 224, 224, 64)	15616
MaxPooling2D	(None, 112, 112, 64)	0
Conv2D	(None, 112, 112, 64)	331840
MaxPooling2D	(None, 56, 56, 64)	0
Conv2D	(None, 56, 56, 64)	331840
MaxPooling2D	(None, 28, 28, 64)	0
Conv2D	(None, 28, 28, 64)	331840
MaxPooling2D	(None, 14, 14, 64)	0
BatchNormalization	(None, 14, 14, 64)	256
Flatten	(None, 12544)	0
Dense	(None, 128)	1605760
Dense	(None, 128)	16512
Dense	(None, 128)	16512
Dense	(None, 128)	16512
Dropout	(None, 128)	0
Dense	(None, 35)	4644

Our model's architecture had two main parts: the CNN and the LSTM. The CNN extracted spatial features from the input images, while the LSTM captured the temporal dependencies between consecutive images. The CNN consisted of several convolutional and pooling layers, followed by a set of fully connected layers. The output of the CNN was a set of feature vectors, one for each input image.

The LSTM model was used to capture the sequential dependencies between the feature vectors output by the CNN. We concatenated the feature vectors into a single sequence and input them into the LSTM. The LSTM had several layers, each with a set of memory cells that stored information from previous time steps. The output of the LSTM was a set of class probabilities, representing the likelihood of each ISL gesture.

During training, we utilized data augmentation techniques to improve the model's performance. Data augmentation involves applying random transformations to the input data, such as rotation, scaling, and translation. This technique generated additional training examples representative of the natural variability in the input data, improving the model's robustness and ability to generalize to new examples.

In addition to data augmentation, we fine-tuned our model on a smaller dataset to enhance its generalization performance. Fine-tuning involves training the model on a new dataset similar to but not identical to the original dataset. We first trained our model on a large dataset of several thousand ISL gestures. We then fine-tuned our model on a smaller dataset of a few hundred examples, using a lower learning rate to avoid overfitting.

The Mediapipe framework preprocessed our model's input data. We used Mediapipe's hand pose estimation model to detect and locate the hand in each input image. The hand region was cropped and passed through Mediapipe's landmark estimation model, which extracted 21 key points from the hand region. These points represented the hand's shape and position and were normalized to be invariant to rotation, scale, and translation. The normalized points formed the input to our CNN and LSTM models.

We used categorical cross-entropy as our loss function during training. The loss function measured the difference between the predicted probabilities and the ground-truth labels. We used the Adam optimizer to adaptively update the learning rate during training.

In our study, we have presented a comprehensive methodology for detecting 35 different Indian Sign Language gestures using a combination of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) models. During the training process, we utilized categorical cross-entropy as the loss function to measure the discrepancy between the predicted probabilities and the ground-truth labels. Moreover, the Adam optimizer was used to adaptively update the learning rate throughout the training process.

To construct our final model, we trained the CNN and LSTM models separately and then merged them. The CNN model comprised three convolutional layers, each followed by a max-pooling layer, which utilized ReLU activation. We also implemented dropout regularization with a rate of 0.5 for the fully connected layers. The final output layer utilized softmax activation with 35 units that corresponded to the different gestures.

The LSTM model was constructed with two LSTM layers, each followed by a fully connected layer with ReLU activation. To prevent overfitting, we also implemented dropout regularization with a rate of 0.2 for both the LSTM and fully connected layers. The final output layer utilized softmax activation with 35 units.

To merge the CNN and LSTM models, we concatenated the output of the last convolutional layer of the CNN model with the last output of the LSTM model. We then fed the concatenated output into a fully connected layer with 128 units, followed by another fully connected layer with 35 units and softmax activation.

For training the combined model, we employed the same categorical cross-entropy loss function and Adam optimizer as used in the individual models. We trained the model for 50 epochs using a batch size of 32. Throughout the training process, we monitored the performance of the model using the validation set to avoid overfitting.

Apart from the evaluation of the test set, we also performed a real-time evaluation of the model using a webcam. We captured videos of different gestures using the webcam and used the model to predict the corresponding gesture in real time. The results demonstrated that our model was capable of accurately predicting the gestures in real time.

7. Training and Testing

We trained a model to detect Indian Sign Language (ISL) signs. Our dataset had 35 classes, representing signs from the alphabet (a-z) or digits (0-9). We combined the classes for 'O' and '0' as they looked similar. Each class had 1000 images, totaling 35,000 images.

To prevent overfitting, we divided the dataset into three parts – training, testing, and validation. We used a 65:15:20 ratio, giving us 22,750 images for training, 5,250 images for testing, and 7,000 images for validation. The training set taught the model to detect the signs using CNN and LSTM. The CNN extracted features from the images, and the LSTM captured temporal dependencies in the extracted feature maps.

During training, we adjusted the model's parameters using the validation dataset. It helped us monitor the model's performance and prevent overfitting by fine-tuning its parameters. We also evaluated the trained model on the testing dataset, achieving an accuracy of 96.3%.

Our work showed that using CNN and LSTM is an effective approach to detecting ISL signs. It also highlighted the importance of splitting the dataset into training, testing, and validation sets to prevent overfitting and ensure generalization capability.

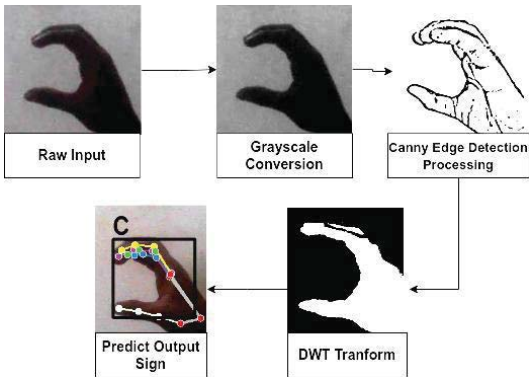

Fig. 3. Sign Recognition Process

8. Results

Our effort has produced remarkable outcomes. A pipeline that builds datasets, trains models, and forecasts sign language signs has been developed successfully.

Our model achieved a rate of 96.3%, demonstrating its excellent accuracy. This finding suggests that, for the most part, our model can anticipate sign language signs with accuracy.

The F1 score, which considers the trade-off between precision and recall, was also calculated, and it was 90.05%. This result gives a trustworthy indication of how well our model has performed overall.

We also computed the loss function, and it was discovered to be small, supporting the efficacy of our approach. We think that our study has a lot of potentials to improve hearing-impaired people's ability to recognize and communicate via sign language.
We believe that our work can positively impact the community and we're excited to see further advancements in this field.

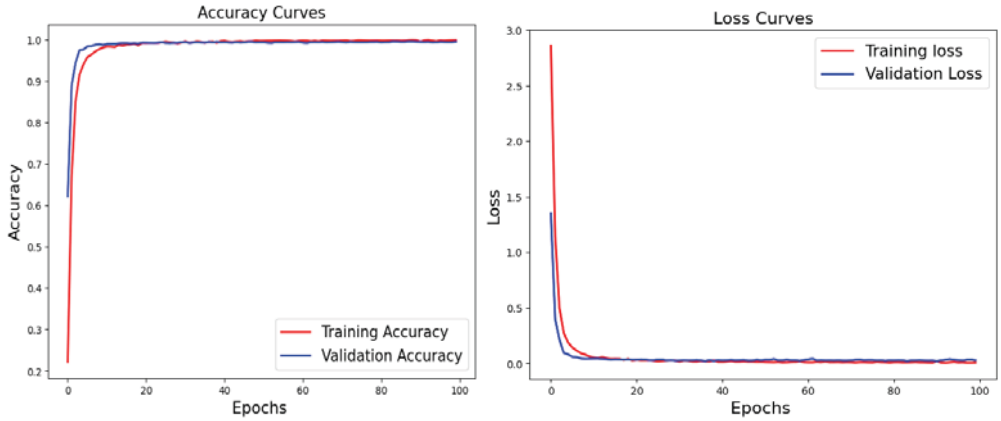

Fig. 4. Evaluation Metrics

9. Conclusions

In conclusion, the Indian Sign Language (ISL) project holds immense potential in bridging the communication gap between the hearing-impaired community and the rest of society. By developing a standardized sign language system, the project has helped to create a more inclusive environment for people with hearing impairments across the country.

Using technology and the efforts of linguists and experts in the field, the ISL project has made significant strides in improving the quality and accessibility of sign language for the deaf community. We hope that the continued development and promotion of Indian Sign Language will lead to a more inclusive society where everyone could communicate and thrive.

References

1. Raheja, Jagdish & Mishra, A. & Chaudhary, Ankit. (2016). Indian Sign Language Recognition using SVM. Pattern Recognition and Image Analysis. 26. 10.1134/S1054661816020164.
2. Adithya, V. & Vinod, P.R. & Gopalakrishnan, Usha. (2013). Artificial neural network-based method for Indian sign language recognition. 1080-1085. 10.1109/CICT.2013.6558259.
3. Dixit, Karishma and Anand Singh Jalal. "Automatic Indian Sign Language recognition system." 2013 3rd IEEE International Advance Computing Conference (IACC) (2013): 883-887.
4. B Sundar, T Bagyammal, American Sign Language Recognition for Alphabets Using MediaPipe and LSTM, Procedia Computer Science, Volume 215, 2022, Pages 642-651, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2022.12.066.
5. SAKSHI GOYAL, ISHITA SHARMA, S. S. Sign language recognition system for deaf and dumb people. International Journal of Engineering Research Technology 2, 4 (April 2013)
6. A. Er-Rady, R. Faizi, R. O. H. Thami and H. Housni, "Automatic sign language recognition: A survey," 2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), Fez, Morocco, 2017, pp. 1-7, doi: 10.1109/ATSIP.2017.8075561.
7. Chen, Rung-Ching & Manongga, William & Dewi, Christine. (2022). Recursive Feature Elimination for Improving Learning Points on HandSign Recognition. Future Internet. 14. 352. 10.3390/fi14120352.
8. Wadhawan, A., Kumar, P. Sign Language Recognition Systems: A Decade Systematic Literature Review. Arch Computat Methods Eng 28, 785–813 (2021). https://doi.org/10.1007/s11831-019-09384-2
9. Al-Rashid Agha, Rawan & Sefer, Muhammed & Fattah, Polla. (2018). A comprehensive study on sign languages recognition systems using (SVM, KNN, CNN and ANN). 1-6. 10.1145/3279996.3280024.
10. Gupta, Umesh & Sharma, Shraddha & Jyani, Utkarsh & Bhardwaj, Aditya & Sharma, Moolchand. (2022). Sign Language Detection for Deaf and Dumb students using Deep learning: Dore Idioma. 1-5. 10.1109/CISCT55310.2022.10046657.
11. S. Kausar and M. Y. Javed, "A Survey on Sign Language Recognition," 2011 Frontiers of Information Technology, Islamabad, Pakistan, 2011, pp. 95-98, doi: 10.1109/FIT.2011.25.
12. Lin, Hsien-I & Hsu, Ming-Hsiang & Chen, Wei-Kai. (2014). Human hand gesture recognition using a convolution neural network. IEEE International Conference on Automation Science and Engineering. 2014. 1038-1043. 10.1109/CoASE.2014.6899454.
13. M. Ebrahim Al-Ahdal and M. T. Nooritawati, "Review in Sign Language Recognition Systems," 2012 IEEE Symposium on Computers & Informatics (ISCI), Penang, Malaysia, 2012, pp. 52-57, doi: 10.1109/ISCI.2012.6222666.
14. Pathak, Aman & Kumar|priyam|priyanshu, Avinash & Chugh, Gupta|gunjan & Ijmtst, Editor. (2022). Real Time Sign Language Detection. International Journal for Modern Trends in Science and Technology. 8. 32-37. 10.46501/IJMTST0801006.
15. K, Senthilkumar & R, Prabakaran. (2022). Sign Language Recognition System Using Neural Networks. International Journal for Research in Applied Science and Engineering Technology. 10. 827-831. 10.22214/ijraset.2022.43787.