Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

# Real-Time Indian Sign Language Detection

**Chhavi Pawan Bothra**
Information Technology
Zagdu Singh Charitable Trust's Thakur Polytechnic
Mumbai, India

**Prachi Ravindra Salunkhe**
Information Technology
Zagdu Singh Charitable Trust's Thakur Polytechnic
Mumbai, India

**Ifra Imitiyaz Shaikh**
Information Technology
Zagdu Singh Charitable Trust's Thakur Polytechnic
Mumbai, India

**Rajas Lalit Patil**
Information Technology
Zagdu Singh Charitable Trust's Thakur Polytechnic
Mumbai, India

The research paper is organised as follows:

| Main Section | Step No. | Step Name | Description |
|---|---|---|---|
| Abstract | - | Abstract | Brief overview of the entire project |
| Methodology | 1 | Introduction | Problem definition and background |
| Methodology | 2 | Literature Review | Study of existing methods |
| Methodology | 3 | Proposed Solution | Overall approach to solve the problem |
| Methodology | 3.1 | Data Acquisition | Collecting dataset |
| Methodology | 3.2 | Image Preprocessing | Cleaning and preparing images |
| Methodology | 3.3 | Feature Extraction | Extracting important features |
| Methodology | 3.4 | Dataset Splitting | Dividing into training and testing data |
| Methodology | 3.5 | Classification | Applying machine learning model |
| Methodology | 3.6 | Evaluation Metrics | Measuring performance |
| Results | 4 | Results | Output and analysis |
| Conclusion | 5 | Conclusion | Summary |
| References | 6 | References | Sources used |

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

**Abstract**

This paper basically focuses on recognizing the Indian Sign Language which is used by deaf and mute people to communicate by making the hand gestures and movements. The limitations faced by the current deaf and mute people is they are unable to convey the message to normal person because he doesn't know the signs exactly. So, here our model can be used serving as the middlemen between them. The objective is to develop a model which will detect the gestures in real time in various diverse conditions. In order to consider variations, we had collected the data manually in the different backgrounds and also, we had considered the skin tones and many other factors in this so the model can predict well in real-time environment. CNN based deep learning algorithm is used giving out the accuracy of 90.20%. The size of datasets is roughly around 23,000 in size. This model will basically bridge the communication gap between deaf and normal person resulting in the proper communication.

## 1. Introduction

According to World Health Organization (WHO) there are approximately 63 million deaf and mute people which constitutes 6.3% of the population. Sign language plays an important role in their daily life to communicate. They totally rely on sign language, but also the language is not known to everyone creating a huge communication gap. However mostly the technological advancements is done on ASL which basically stands for American Sign Language (the signs varies region to region) but the ISL still lacks behind and remain underrepresented. Although some sort of models exists but they don't consider as much variations, here comes our model in existence.

The problem can be solved till the certain point with the help of deep learning and computer vision. So, we have used CNN (Convolutional Neural Network) which is a good option for image processing. We had tried multiple models and got the highest accuracy in CNN and also, we had gone through several research papers to find out the suitable model. In our model we are using open cv for the camera access and the media pipe to plot the 21 coordinates of our hand based on this the models predict the gesture and gives the corresponding output to it.

The sign language approach can be basically considered into three (Character-level, Word-level and Sentence-level):

- **Character-level**: It can be referred to as the conversion of gesture and movements into character.
- **Word-level:** It refers to the conversion of gesture and movements into word.
- **Sentence-level**: It refers to converting the gesture or signs into sentence.

The approach we are following is character-level based which includes alphabets and numbers.

As mentioned earlier Sign language is not same across it varies region to region therefore the models which are trained on ASL will not predict the signs for the ISL. Both are sign language are fundamentally different. ISL consists of the two-hand gestures in it making it more complex and also it is hard to recognize, that's the reason we chose ISL.

The image below shows the Technology Stack used:


**Fig 1. Technology Stack**

The primary objective is:

- to collect as much dataset as possible.
- to achieve the highest accuracy among other existing models.

## 2. Literature Review

Many studies are being conducted in this field which uses the deep learning technique.

- Sharma et al. (2020) he made the sign language detection system which used the SVM (Support Vector Machine). The accuracy was average because the model was not considering the diverse or variations in the sign gestures.
- Patil and Desai (2021) they both also proposed the system which used the CNN the model showed the better accuracy as compared to that of SVM but the only prerequisite was the availability of the large dataset and the resources for training purposes.
- Verma et al. (2022) he followed the deep learning approach also the model performed well but it was facing major issues in real world conditions due to variations.

Also, after going through many research papers most of the research papers currently available are based on ASL which is totally a different thing as compared to that of ISL. ISL consists of the more complex gestures which are two-handed gestures or signs. Also, we can conclude that using CNN (better for image processing) based model it gives improved accuracy as compared to that of others and also considering diverse variations in the lighting and skin tones.

## 3. Proposed solution

The system is designed to identify and sort Indian Sign Language (ISL) signs using a machine learning model. The system follows several steps to complete this task. These steps include gathering data, cleaning and preparing the images, extracting important details from the images, dividing the data into training and testing sets, training the model to recognize signs, and evaluating the model's performance.

Initially, there is no dataset available on the internet (like on open-source dataset websites Kaggle, Hugging Face, etc.), so we decided to collected it manually. We collect a set of images of ISL gestures manually under many different real-life situations (like with different skin tones, age groups, backgrounds, angles, etc.). Once the images are collected, they are cleaned and improved to make them clearer and free

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

from unwanted details.

After reprocessing, the most important details are extracted, which are then used to train the model that can recognize the sign. After the model is trained, it is tested using standard measures to see how effective it is.

### 3.1 Data Acquisition

In this research, images were manually acquired to ensure diversity of images and work well in real life. We collected images for the letters A to Z and the numbers 0 to 9, excluding the number 6. The dataset reflects variations in backgrounds (including simple, complex, indoor, and outdoor environments), skin tones, age groups, and lighting conditions as shown in Fig 1. Around 23,000 images were collected, making the dataset both large and varied. This variety helps the model work well in different situations. The images were taken with regular cameras and sorted into groups based on the signs shown.

**Fig.2**

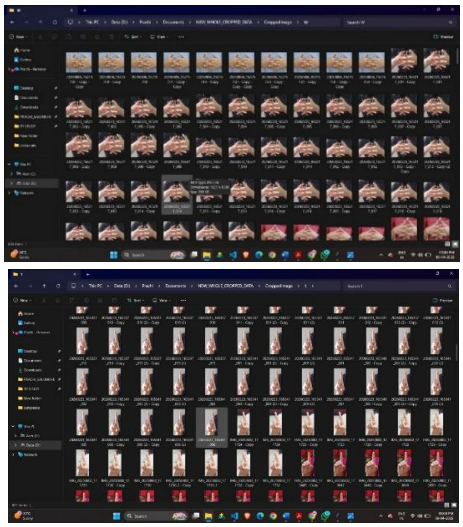

**Fig 3. Dataset**

### 3.2 Image Reprocessing

The performance of deep learning models is greatly affected by how good and consistent the input data is. Because the Indian Sign Language (ISL) dataset collected has a lot of differences in background, lighting, hand position, and people involved, a detailed pre-processing process was created to make all the input data the same. All the preprocessing is done using the PyTorch library.

#### 3.2.1 Image Standardization

To make sure all images have the same size for the Convolutional Neural Network (CNN), every image is changed to 128 x 128 pixels. This step makes the processing faster and works well with the network design.

Resizing also helps in:

- Using less memory.
- Keeping all the data consistent.
- Making training faster.

#### 3.2.2 Data Augmentation for Generalization

To help the model perform better and avoid learning too much and make it more reliable from the training images, we use many data augmentation methods on the training data. These methods create variations (see in Fig.2) that are like those of what happens in real life and add more variety in the data without extending the amount of dataset.

We used several techniques to make the training data more varied:

- **Random Rotation (±15 degrees):** Helps the model understand hand signs regardless of rotation.
- **Random Horizontal Flipping (probability = 0.5):** Makes the model work well with mirrored signs.
- **Color Jittering (Changing Brightness and Contrast):** Helps the model handle different lighting situations.
- **Random Resizes Cropping (scale: 0.8 to 1.0):** Introduces changes in size and position, making the model better at handling slight movements.'

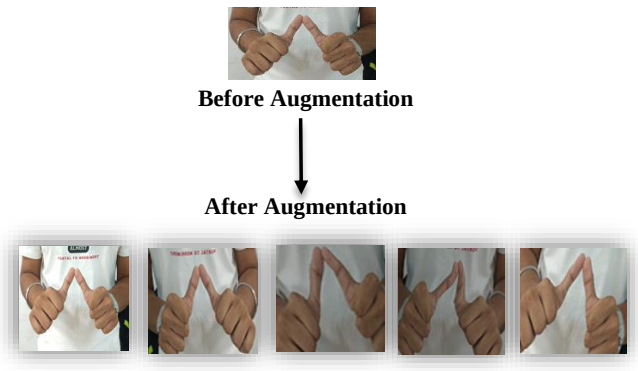


**Fig.4 Data Augmentation**

These changes help the model perform better in real situations that it hasn't seen before.

#### 3.2.3 Pixel Normalization

All images were converted into a special format called tensors.

Then, we adjust the pixel values using specific numbers [0.5,0.5,0.5] so they fall within the range of -1 to 1. This helps

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

the model train faster and more stably.

**3.2.4 Pre-processing for Training vs. Testing**

It's important to note that:

- Training images go through both augmentation and normalization.
- Validation and test images only get resized and normalized.

This ensures the test results are fair and show how the model performs in real life.

**3.2.5 Impact of Preprocessing**

Pre-processing helps in several ways: -

- Reducing unwanted details and making the data consistent.
- Improving the accuracy of the model.
- Making the model better at handling different real-life situations.

**3.3 Feature Extraction**

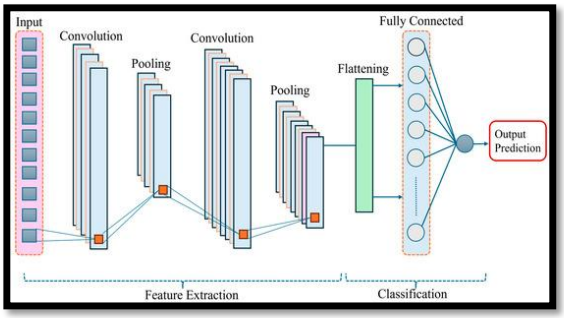


**Fig 5. CNN Layers**

Feature extraction is a key part of the system, as it helps the model understand and tell apart different ISL signs. In our project, feature extraction is done automatically using CNN in PyTorch.

**3.3.1 Hierarchical Feature Learning**
The CNN learns features in a step-by-step way:

- Low-level feature: edges, corners, textures.
- Mid-level feature: curves, outlines, and shapes of signs.
- High-level feature: full representation of signs.

This step-by-
step process helps the model build a deeper understanding of the images.

**3.3.2 Convolutional Operations**

Each convolutional layer uses multiple filters (kernels) that move across the image. These filters detect patterns and activate when they find something interesting.

Mathematically, convolution can be expressed as:

$$\textit{Feature Map} = \textit{Input} \otimes \textit{Kernels}$$

Where the kernel finds things like edges and textures in the image.

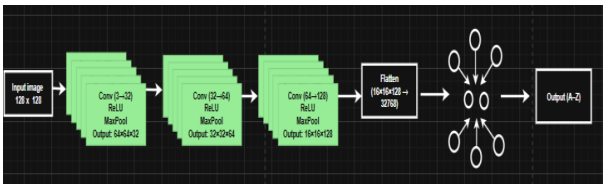


**Fig 6. Feature Extraction**

**3.3.3 Deep Feature Representation**

The CNN has three layers with increasing numbers of filters:

- Layer 1 (32 filters): Captures basic visual details.
- Layer 2 (64 filters): Learns intermediate features.
- Layer 3 (128 filters): Extracts complex features related to signs.

As the layers get deeper,
the network finds more abstract and meaningful information.

**3.3.4 non-linearity using ReLU**

After each convolution operation, the ReLU (Rectified Linear Unit) activation function is used:

$$F(x) = \max(0, x)$$

This adds Non-Linear behavior, allowing the networks to model more complicated relationships between inputs and outputs.

**3.3.5 Spatial Downsampling Using Max Pooling Max**

Max pooling layers reduce the size of feature maps while keeping important information.
This helps with:

- Lowering computational load.
- Preventing overfitting.
- Marking the model flexible to position changes.

The size of the feature maps reduces as follows:

$$128 \times 128 \rightarrow 64 \times 64 \rightarrow 32 \times 32 \rightarrow 16 \times 16$$

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

#### 3.3.6 Feature Vector Generation

After the convolution and pooling steps, the feature maps are turned onto a one-dimensional feature vector. This vector represents the features learned from the image and is used as input for fully connected layers.

#### 3.3.7 Regularization using Dropout

A dropout layer with a 0.5 probability is used in the fully connected layers. This method randomly stops some neurons from working during training, which:

- Stops overfitting.
- Makes the model more general.
- Encourages the network to learn stronger features.

Using CNN for feature extraction, we adapt real-world data variations; there is no need for manual feature selection; it provides accuracy compared to traditional methods; and the last one is to learn detailed and useful features.

### 3.4 Dataset Splitting

To ensure reliable training and unbiased evaluation, the dataset is partitioned into three distinct subsets: training, validation, and testing. In this study, a split ratio of 70:10:20 is adopted, where 70% of the data is used for training, 10% for validation, and 20% for testing.

The training set is utilized to learn the underlying patterns and feature representations from the input data. The validation set is employed during the training process to fine-tune hyperparameters, monitor model performance, and prevent overfitting by providing intermediate feedback. The testing set, which remains completely unseen during both training and validation phases, is used for final performance evaluation.

To maintain consistency across all subset's, stratified sampling is applied to preserve the distribution of different classes. This structured division enhances the model's ability to generalize effectively and provides a more accurate estimate of its performance in real-world applications.

### 3.5 Classification

In this work, a Convolutional Neural Network (CNN) is implemented using the PyTorch deep learning framework for the classification of Indian Sign Language gestures. The model is designed to automatically learn discriminative features from input images and accurately assign them to their respective classes.

The proposed architecture consists of three convolutional blocks followed by fully connected layers. Each convolutional block includes a convolutional layer with a kernel size of 3×3, stride of 1, and padding of 1, ensuring preservation of spatial dimensions. The first convolutional layer transforms the input image from 3 channels to 32 feature maps, followed by subsequent layers increasing the depth to 64 and 128 feature maps, respectively. Each convolutional operation is followed by a Rectified Linear Unit (ReLU) activation function to introduce non-linearity.

Max-pooling layers with a kernel size of 2×2 is applied after each convolutional block to reduce spatial dimensions and computational complexity. This progressively reduces the feature map size from **128×128 to 64×64**, then to **32×32**, and finally to **16×16.**

The extracted feature maps are then flattened into a one-dimensional vector and passed through fully connected layers. A dense layer with **512 neurons** is used, followed by a ReLU activation and a dropout layer with a dropout rate of 0.5 to reduce overfitting. The final output layer maps the features to the number of gesture classes using a linear transformation.

For training, the Cross-Entropy Loss function is employed to measure the difference between predicted and actual class labels. The model parameters are optimized using the Adam optimizer with a learning rate of 0.001. The training process is conducted over 25 epochs, during which the model learns by minimizing the loss through backpropagation.

The dataset is processed in batches using data loaders, enabling efficient training. The model is trained on either a GPU or CPU depending on availability, which enhances computational efficiency. During each epoch, both training and validation phases are performed to monitor model performance. Accuracy is computed for both phases to evaluate learning progress.

To ensure model persistence, the trained weights are saved after each epoch, along with a final saved model file for deployment. This CNN-based approach enables robust and efficient classification of ISL gestures, achieving high accuracy and suitability for real-time applications.

#### 3.5.1 Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN) is a specialized deep learning architecture designed for processing image data. It consists of multiple layers that progressively extract meaningful features from raw input images.

The convolutional layers apply learnable filters to capture local patterns such as edges, textures, and shapes. Activation functions like Rectified Linear Unit (ReLU) introduce non-linearity, enabling the network to model complex relationships. Pooling layers are used to reduce spatial dimensions, which helps in lowering computational cost and improving generalization.

CNNs eliminate the need for manual feature extraction, as they automatically learn hierarchical representations directly from the input data. This makes them highly effective for image classification tasks such as sign language recognition.

#### 3.5.2 Training the Model

The CNN model is trained using labeled image data, where each image corresponds to a specific sign language class. During training, the model learns to map input images to their

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

correct labels by minimizing a loss function.

In this implementation, the Cross-Entropy Loss function is used, as it is well-suited for multi-class classification problems. The optimization of model parameters is performed using the Adam optimizer with a learning rate of 0.001, which ensures efficient and stable convergence.

The training process is carried out over multiple epochs, where the entire dataset is passed through the network several times. The data is processed in batches to improve computational efficiency. After each iteration, the model weights are updated using backpropagation.

To evaluate learning progress, both training and validation accuracy are monitored during each epoch. This helps in detecting overfitting and ensures that the model generalizes well to unseen data.

### 3.5.2 CNN Architecture

The proposed CNN architecture consists of three convolutional layers followed by fully connected layers for classification.

The first convolutional layer takes a 3-channel input image and produces 32 feature maps using a 3×3 kernel. The second and third convolutional layers increase the number of feature maps to 64 and 128, respectively. Each convolutional layer is followed by a ReLU activation function and a max-pooling layer of size 2×2, which reduces spatial dimensions progressively.

After feature extraction, the output is flattened into a one-dimensional vector and passed through a fully connected layer with 512 neurons. 1.A dropout layer with a rate of 0.5 is applied to reduce overfitting. Finally, an output layer maps the features to the total number of gesture classes.

This architecture effectively balances computational efficiency and classification performance, making it suitable for real-time gesture recognition tasks.

### 3.6 Evaluation Matrix

To assess the performance of the proposed model, several evaluation metrics are used. These metrics provide a comprehensive understanding of the classification performance across different gesture classes.

### 3.6.1 Accuracy

Accuracy measures the overall correctness of the model by calculating the ratio of correctly predicted samples to the total number of samples. The proposed model achieves high accuracy, indicating effective learning and classification capability.

**Formula:** $\frac{TP+TN}{TP+TN+FP+FN}$

### 3.6.2 Precision

Precision reflects the reliability of the model's positive predictions. It indicates how many of the predicted instances for a particular class are actually correct. High precision implies a low false positive rate.

**Formula:** $\frac{TP}{TP+FP}$

### 3.6.3 Recall

Recall measures the model's ability to correctly identify all relevant instances of a class. A high recall value indicates that the model successfully captures most of the actual positive samples.

**Formula:** $\frac{TP}{TP+FN}$

### 3.6.4 F1-Score

The F1-score is the harmonic mean of precision and recall. It provides a balanced evaluation metric, especially useful when there is an uneven distribution of classes.

**Formula:** $\frac{2\cdot \mathrm{Pr}ecision\cdot Recall}{\mathrm{Pr}ecision+Recall}$

Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 0.98 | 1.00 | 0.99 | 99 |
| 2 | 1.00 | 0.97 | 0.98 | 98 |
| 3 | 1.00 | 0.97 | 0.98 | 99 |
| 4 | 0.97 | 1.00 | 0.99 | 99 |
| 5 | 1.00 | 1.00 | 1.00 | 99 |
| 7 | 1.00 | 1.00 | 1.00 | 96 |
| 8 | 1.00 | 1.00 | 1.00 | 102 |
| 9 | 1.00 | 1.00 | 1.00 | 99 |
| A | 1.00 | 1.00 | 1.00 | 108 |
| B | 1.00 | 1.00 | 1.00 | 104 |
| C | 0.99 | 1.00 | 1.00 | 102 |
| D | 1.00 | 0.99 | 0.99 | 99 |
| E | 1.00 | 1.00 | 1.00 | 99 |
| F | 1.00 | 0.99 | 1.00 | 104 |
| G | 1.00 | 1.00 | 1.00 | 99 |
| H | 1.00 | 1.00 | 1.00 | 99 |
| I | 1.00 | 1.00 | 1.00 | 108 |
| J | 1.00 | 1.00 | 1.00 | 99 |
| K | 1.00 | 0.98 | 0.99 | 99 |
| L | 1.00 | 1.00 | 1.00 | 102 |
| M | 0.99 | 1.00 | 1.00 | 113 |
| N | 0.98 | 1.00 | 0.99 | 99 |
| O | 1.00 | 1.00 | 1.00 | 104 |
| P | 0.99 | 1.00 | 1.00 | 103 |
| Q | 1.00 | 0.99 | 0.99 | 99 |
| R | 1.00 | 0.98 | 0.99 | 99 |
| S | 0.98 | 1.00 | 0.99 | 99 |
| T | 0.98 | 0.99 | 0.98 | 99 |
| U | 1.00 | 1.00 | 1.00 | 99 |
| V | 1.00 | 1.00 | 1.00 | 99 |
| W | 1.00 | 0.99 | 1.00 | 130 |
| X | 0.99 | 1.00 | 1.00 | 102 |
| Y | 1.00 | 1.00 | 1.00 | 104 |
| Z | 1.00 | 1.00 | 1.00 | 102 |
| accuracy | | | 1.00 | 3464 |
| macro avg | 1.00 | 1.00 | 1.00 | 3464 |
| weighted avg | 1.00 | 1.00 | 1.00 | 3464 |

**Fig 7. Classification Report**

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

### 3.6.5 Confusion Matrix

A confusion matrix is used to visualize the performance of the classification model. It presents a tabular representation of actual versus predicted class labels. 5.The diagonal elements represent correct predictions, while off-diagonal elements indicate misclassifications.

This matrix helps in identifying specific classes where the model performs well and those where improvement is required.

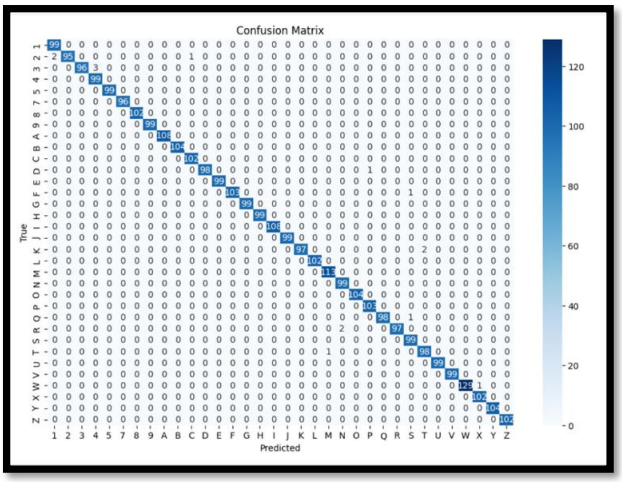


**Fig 8. Confusion Matrix**

## 4. RESULTS AND DISCUSSION

The developed Indian Sign Language (ISL) recognition model was evaluated using training logs, confusion matrix, classification report, and real-time prediction output.

### 4.1 Training Performance

The model was trained for 22 epochs on CPU. At the starting stage, the model showed a training accuracy of 32.35% and validation accuracy of 77.86%, which indicates that the model had not yet fully adapted to the dataset.

As the epochs increased, both training and validation performance improved continuously. The model gradually reduced its errors and started identifying gesture patterns more effectively. By the final epoch, the training accuracy reached 93.95%, while the validation accuracy increased to 99.53%.

At the same time, the training loss decreased from 2.3527 to 0.1935, and validation loss reduced to 0.0222. This consistent reduction in loss values shows that the model is learning properly and making fewer mistakes during prediction.

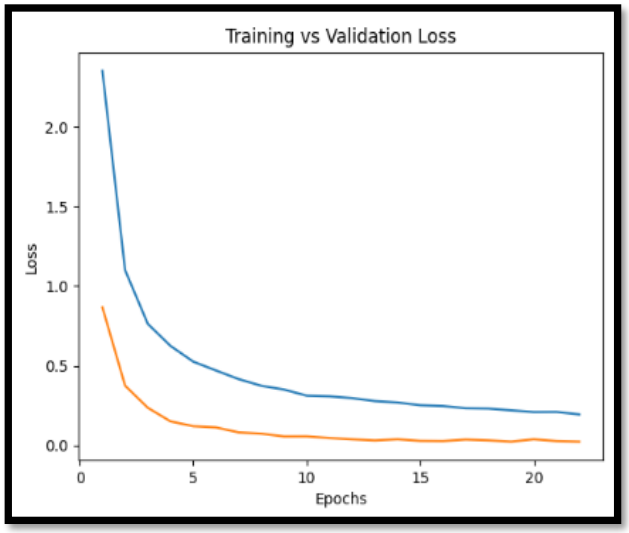


**Fig 9. Training and Validation Performance**


**Fig 10. Training and Validation loss**

### 4.2 Confusion Matrix Analysis

The confusion matrix provides a clear view of how well the model performs for each gesture class.

From the obtained matrix, it can be observed that most of the values are concentrated along the diagonal. This means that for most inputs, the predicted class matches the actual class. Only a very small number of values appear outside the diagonal, indicating very few incorrect predictions.

This pattern shows that the model can correctly differentiate between multiple gesture classes with high accuracy and very low confusion as shown above in confusion matrix.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

### 4.3 Classification Report Analysis

The classification report presents the precision, recall, and F1-score for each class.

From the results, it can be seen that almost all classes have values close to 1.00, which indicates very high performance. The overall accuracy of the model is approximately 1.00 on 3464 samples.

The macro average and weighted average are also equal to 1.00, showing that the model performs consistently across all classes without bias.

Small variations can be seen in a few classes where values are slightly below 1.00, but these differences are minimal and do not significantly affect the overall result.

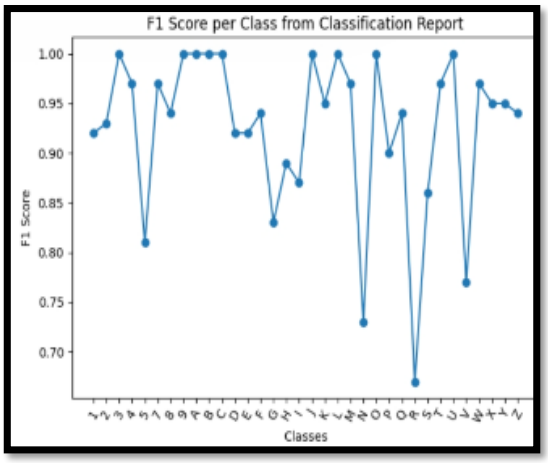


**Fig 11. F1-score and Classes**

### 4.4 Real-Time Prediction

The system was also tested using real-time input. The model successfully predicted the gesture along with a confidence score.

For example, the model predicted the class “3” with a confidence value of 0.97%, which shows that the model is highly confident in its prediction.

This confirms that the system is capable of working in real-time scenarios


**Fig 12. Real-Time Output**

### 4.5 Final Outcome

Based on all the results, the model performs with very high accuracy and reliability. It is able to correctly recognize almost all gestures with very few errors.

## 5. CONCLUSION

This project presents a system that can recognize Indian Sign Language gestures using a trained deep learning model. The system takes hand gesture input and predicts the corresponding output, making it easier for users to understand sign language.

The main objective of this work is to reduce the communication gap between normal individuals and people who are deaf and mute. In many situations, communication becomes difficult because sign language is not widely understood. This system provides a way to make that interaction simpler by converting gestures into readable form.

From the results obtained, it is clear that the model performs very well. The training process shows steady improvement, and the final accuracy values indicate that the model has learned the gesture patterns effectively. The confusion matrix confirms that most predictions are correct, and the classification report shows that performance is consistent across all classes.

Another important observation is that the model is able to give correct predictions in real-time with high confidence. This makes the system useful not only for testing but also for practical applications.

Overall, the developed system is accurate, stable, and efficient. It can be used as a basic tool to support communication with deaf and mute individuals. With further improvements, it can be expanded to handle more complex gestures and real-world situations.

## 6. REFERENCES


- Indian Sign Language Research and Training Centre (ISL RTC) https://ariv.org/.pdf/2018.10970
- Research papers on Sign Language Recognition https://diyangan.depwd.gov.in/islrtc/seacrh.php
- TensorFlow, Pytorch official docs https://www.tensorflow.org/api_docs