# Sign Speak: Indian Sign Language to Text Convertor

1st Mr. Vikas G Singh , 2nd Ms. Prakriti Jain, 3th Mr. Onkar Bhute, 4th Mr. Shreeyash Bachal, 5th Mr. Shreyas Janbandhu, 6th Mr. Shreyash Kotangale, 7th Mr. Virag Jambhore
1,2,3,4,5,6,7 *Dept. of Computer Science and Engineering GHRCEM* Nagpur, India

**Abstract: Indian Sign Language (ISL) serves as a vital mode of communication for the deaf and hard-of-hearing community in India. However, the lack of widespread understanding of ISL among the general population creates communication barriers. This project presents an "Indian Sign Language to Text Converter," a real-time system designed to bridge this gap by translating ISL gestures into readable text. The system leverages computer vision and machine learning techniques to recognize hand gestures and convert them into corresponding text. Using Python, OpenCV, TensorFlow, and Keras, the project implements a deep learning model for gesture recognition. A custom dataset of ISL gestures was created, involving image preprocessing techniques such as background subtraction, edge detection, and skin color segmentation to enhance accuracy. Real-time video input allows dynamic sign detection, with the system optimized for low-latency performance. The system aims to foster inclusivity and improve communication by providing an accessible tool for ISL users and non-signers alike.**

**Keywords: - Machine Learning Algorithms, RNN, CNN, Indian sign language, Gesture recognition.**

## I. INTRODUCTION

Communication is a fundamental human need, yet millions of deaf and hard-of-hearing individuals face barriers due to the lack of widespread knowledge of sign languages. In India, Indian Sign Language (ISL) is the primary medium of communication for this community. However, the limited proficiency in ISL among the general population creates challenges in day-to-day interactions. This project aims to address this issue by developing a real-time Indian Sign Language to Text Converter, which can translate ISL gestures into text. By employing computer vision techniques and machine learning models, the system is designed to recognize hand gestures and provide an accurate textual representation. The solution integrates advanced tools such as OpenCV for image processing and TensorFlow and Keras for gesture recognition, ensuring real-time performance and accuracy. This project strives to create a bridge between ISL users and non-signers, promoting inclusivity and improving accessibility in communication.

## II. LITERATURE REVIEW

The intersection of technology and social inclusivity has garnered significant attention in recent years, particularly with the rise of digital solutions. This literature review highlights key themes and findings relevant to SignSpeak, focusing on building the bridge between the hearing-impaired community and the abled ones, the importance of social inclusivity, and the role of individual to support it.

[1] Seema Sabharwal, Priti Singla: The research paper focuses on developing a machine learning system that translates Indian Sign Language (ISL) gestures into text. The paper also focuses on a ground-breaking approach to improving communication for deaf and mute individuals by developing a system that translates Indian Sign Language (ISL) gestures into text. By harnessing the power of machine learning, specifically using Convolutional Neural Networks (CNNs), the authors aim to bridge communication gaps in the deaf community.

[2] Akash Kamble, Jitendra Musale, Rahul Chalavade, Rahul Dalvi, Shrikar Shriyal: This study successfully demonstrates a multi-headed CNN that effectively combines image data and hand landmark information to recognize ASL gestures with high accuracy, even in challenging real-world environments. By fusing two input streams, the model overcomes the limitations of traditional single-input CNNs, leading to enhanced classification performance. Techniques such as data augmentation, batch normalization, dropout, and dynamic learning rate adjustment further optimize the model's ability to generalize and perform well in practical scenarios.

[3] Refat Khan Pathan, Munmun Biswas, Suraiya Yasmin, Mayeen Uddin Khandekar, Mohammad Salman, Ahmed A.F. Youssef(2023): This paper introduces a system designed to bridge

communication gaps for the deaf and mute community by translating Indian Sign Language (ISL) gestures into text and voice. Using image processing techniques and LDA for gesture recognition, it delivers high accuracy, with plans to expand its capabilities to include numbers and more complex signs.

[4] , Pooja Bagane, Muskan Thawani, Prerna Singh, Raasha Ahmad, Rewaa Mital, Obsa Amenu, Jebessa: This paper presents an innovative system that translates American Sign Language (ASL) gestures into text using computer vision and machine learning. By leveraging CNNs and Transfer Learning with ResNet-50, the system accurately recognizes hand gestures, enabling individuals with hearing or speech impairments to communicate effectively.

## III. METHODOLOGY

1.Initialization:

The process pipeline starts from the initialization of the operation by setting up the environment used in data acquisition and further operations in the pipeline.

2.Data Acquisition:

In this step, raw data-examples are gathered from various sources, which may include already existing data banks or presently available sensors. The objective of this step is to gather data samples on which the model could be trained during the training stage.

3. Data Preprocessing:

After data gathering is done, preprocessing follows, whose key aim is to produce data that will be uniform and hence ready for training:
- Database Merge: Multiple datasets are merged into one dataset for uniformity.
Encoding Categorical Variables: Gesture label names are encoded in numerical format so the machine learning algorithms can understand.
- Data Scaling and Normalization: Feature values, such as pixel intensities, are scaled to a uniform range (e.g., [0,1]) to prevent large-scale differences between feature magnitudes from biasing the model.

4. Data Splitting:

The preprocessed data split into different sets:
- Training Data: The most significant part of data used to learn by the model.
- Test Data: Reserves a small portion for the evaluation of the model after training.
- Validation Data: Alternatively, yet another set is used for hyperparameter tuning at training, to prevent overfitting.

5. Modeling

At this step, the real training of a machine learning model is done:
- Model Selection : For gesture recognition, spatial features are captured through Convolutional Neural Networks(CNNs), while RNN or LSTMs are used for capturing time series sequences depending on the complexity of data.
- Model Training: The CNN learns spatial patterns, such as the shape of hand or curves, and the RNN learns sequential dependencies in gesture movements.
- Model Evaluation: At various intervals during training, the model is evaluated on validation data to make sure it has learned the concepts it is supposed to learn and to be properly generalizing.

6. Data Visualization

Data visualization aids in tracking the learning process of a model and checks for overfitting/underfitting with the aid of accuracy plots or confusion matrices, etc.

7. Testing

In the case of performance testing on unseen inputs, the measures such as accuracy, precision, recall, and sometimes F1-score are produced by test data.

8. Predictions

After testing, it is implemented in real applications of prediction of new gesture data in sign language recognition or other systems based on proper classification.

This pipeline transforms raw data into actionables at the prediction level while using a structured, spatial, and temporal learning approach via CNNs and RNNs. The workflow is pretty optimized as far as the model's accuracy and performance assessment is concerned.

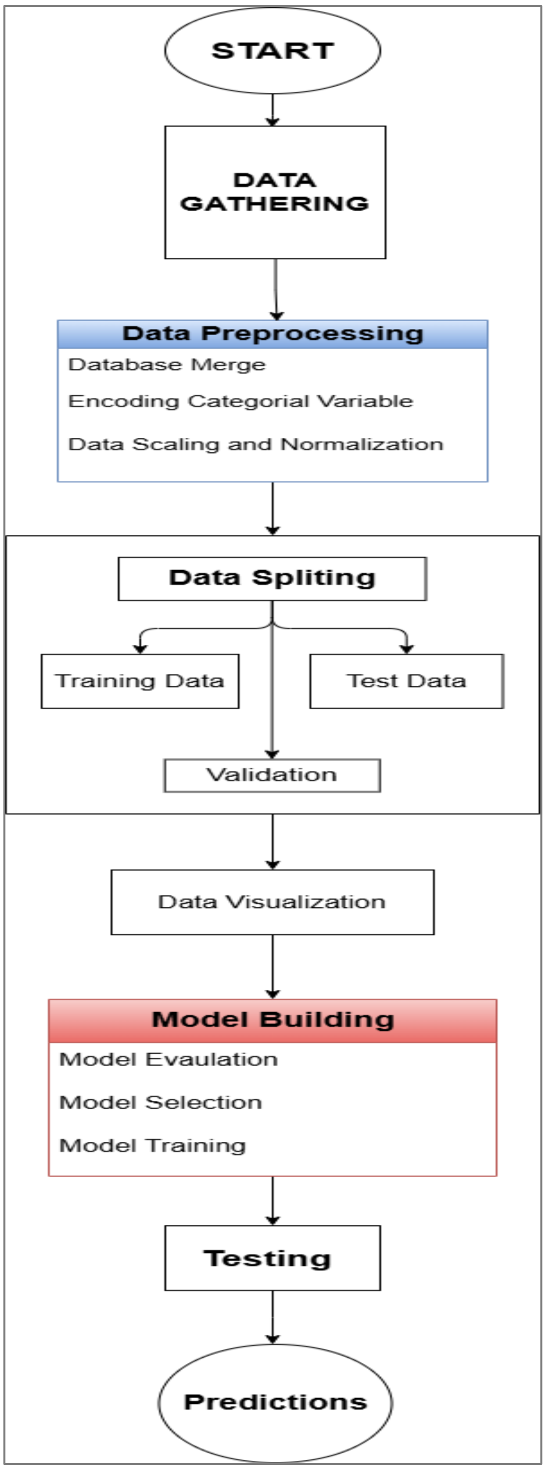

*Fig 1: Block Diagram*

1. Software requirement:

Programming Languages:

- Python: Primary language for implementing CNN, NLP models.

Development Environment:

- Jupyter Notebook / PyCharm / Visual Studio Code: For developing and testing NLP models and integrating them into web UI

Library used: -

OpenCV (Open-Source Computer Vision Library) was employed for image and video analysis, particularly for hand tracking, gesture segmentation, and feature extraction, facilitating efficient processing of sign language gestures. MediaPipe, developed by Google, provided real-time hand and pose tracking, leveraging pre-trained gesture recognition models to identify hand movements and key points critical for sign interpretation.

For deep learning, TensorFlow was the framework of choice due to its scalability in training custom models, from basic hand gestures to complex, dynamic signs. Keras, a high-level API built on TensorFlow, streamlined the process of building and training Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for accurate gesture recognition.

Flask, a lightweight web framework, was utilized to deploy the model in real-time, allowing seamless interaction between the recognition system and user interface. NumPy played a fundamental role in handling large datasets of gesture images, efficiently managing data arrays for computational processing.

Finally, Deep-Translator was integrated to ensure multilingual support, translating recognized signs into multiple languages for enhanced global communication.

2. Technologies:

a. CNN: -

Convolutional Neural Networks (CNNs) are fundamental to deep learning, particularly in image recognition and computer vision. Their architecture, consisting of convolutional layers for feature extraction and pooling layers for dimensionality reduction, allows them to automatically learn spatial hierarchies in data. By stacking layers, CNNs capture patterns from simple edges to complex textures, making them highly effective in recognizing and interpreting images.

Training CNNs typically involves large, labeled datasets and optimization algorithms like stochastic gradient descent (SGD) or Adam. Techniques such as dropout and batch normalization prevent overfitting, ensuring models generalize well to new data. This has led to revolutionary applications, with CNNs excelling in fields like object detection, medical imaging, and autonomous driving, where their ability to automatically identify relevant features outperforms traditional methods.

However, CNNs have limitations. They rely heavily on vast, well-annotated datasets, which are often scarce in specialized fields. Moreover, they struggle with variations in scale, rotation, and orientation of input data, requiring additional preprocessing. The high computational cost of training deep CNNs also limits their deployment on resource-constrained devices.

To address these challenges, transfer learning and lightweight architectures such as MobileNet and SqueezeNet have been developed, allowing CNNs to be used in low-resource environments. Integration with other deep learning models, like Recurrent Neural Networks (RNNs) and Transformer architectures, is also being explored to handle more complex data types.

CNN ARCHITECTURE: -

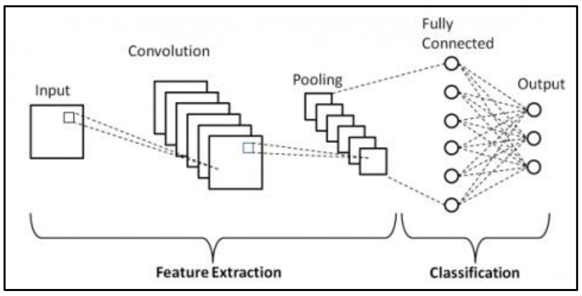

*Fig 2: Architecture of Convolutional Neural Networks*

b. RNN: -

Recurrent neural networks have marked a significant advancement in processing sequential data. This approach has gained popularity across various fields, including natural language processing, time series forecasting, and speech recognition, due to its unique ability to retain contextual information through recurrent connections. The discussion here centers on the structure, training methods, and uses of these networks, which have proven superior to earlier, more traditional techniques in several domains.

These networks are specifically designed to process sequential inputs, with a hidden state carrying information from one step to the next. This feature ensures that the sequence of inputs plays a crucial role during processing. Variants, such as LSTMs and GRUs, enhance this functionality by addressing challenges like vanishing and exploding gradients, making the networks more efficient in learning long-term dependencies.

The architecture of these networks is similar to standard neural networks but requires an input layer, one or more recurrent hidden layers, and an output layer. At each time step, the hidden state is updated, enabling the network to retain and utilize information from previous inputs. LSTMs and GRUs add gating mechanisms that regulate the flow of information, allowing the network to learn from long-term dependencies more effectively.

Training typically involves the backpropagation through time (BPTT) algorithm, which adjusts the network weights over multiple time steps to enable learning from sequences of variable length. The training process aims to minimize a loss function, with cross-entropy loss being commonly used in classification tasks.

Evaluating the performance of these models often involves metrics such as accuracy, precision, recall, and the F1-score, which provide insights into the predictive capabilities and overall reliability of the trained model. These metrics help assess how well the model generalizes to unseen data and handles complex patterns in sequential inputs.

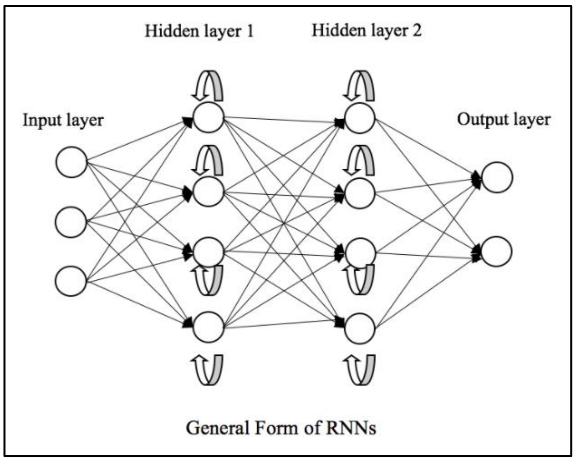

*Fig 3: Architecture of Recurrent Neural Networks*

Integrated CNN-RNN code: -

```
IMPORT libraries
LOAD dataset
PREPROCESS images
SPLIT dataset
INITIALIZE cnn_model
ADD Convolutional layers
ADD Pooling layers
ADD Flatten layer
ADD Fully connected layers
ADD Output layer
COMPILE cnn_model
FIT cnn_model
SAVE cnn_model
INITIALIZE rnn_model
ADD Embedding layer
REPEAT
ADD RNN layer
ADD Dense layer
COMPILE rnn_model
FIT rnn_model
SAVE rnn_model
INITIALIZE Flask app
DEFINE home route
DEFINE processing route:
RECEIVE image
PREPROCESS image
PREDICT gesture
TRANSLATE to text
 RETURN sentence
 RUN Flask app
 IF issues
COLLECT more data
RE-TRAIN models
END
```

## IV. RESULTS

High Accuracy in Gesture Recognition
The system excels at recognizing both static and dynamic Indian Sign Language (ISL) gestures, thanks to rigorous model training and advanced data augmentation techniques. This ensures a robust and reliable recognition system.

Real-Time Translation Capability
The system operates in real-time, translating ISL gestures into text with minimal delay, providing a seamless communication experience. It is optimized for low-latency performance, even on devices with limited computational power.

Custom Dataset Creation
A custom, meticulously annotated ISL gesture dataset was developed for this project, contributing significantly to the machine learning model's high recognition accuracy.

Advanced Image Preprocessing Techniques
The system employs techniques like background subtraction, edge detection, and skin colour segmentation to enhance gesture clarity, improving recognition accuracy even in challenging visual environments.

Dynamic Gesture Detection
In addition to static gestures, the system can recognize dynamic gestures, expanding its use in real-world applications.

Continuous Learning and Scalability
The machine learning model adapts over time, improving its recognition capabilities with ongoing retraining. Its architecture is designed for scalability, allowing the inclusion of new gestures and languages with ease.

Robustness and Accessibility
The system performs well across diverse environments, thanks to robust preprocessing techniques. This project has the potential to enhance accessibility for the deaf and hard-of-hearing communities, fostering inclusivity and improving communication.

## REFERENCES

Journals:
[1] Optimised Machine Learning- based Translations of Indian sign Language to Text, Seema Sabharwal, Priti Singla (International Journal of Intelligent Engineering and Systems, Vol.16, No.4,2023 DOI:10.22266/ijies2023.0831.32).
[2] Sign language recognition using the fusion of image and hand landmarks through multi-headed convolution neural network, Refat Khan Pathan, Munmun Biswas, Suraiya Yasmin, Mayeen Uddin Khandekar, Mohammad Salman, Ahmed A.F. Youssef (https://doi.org/10.1038/s41598-023-43852-x ww.nature.com/scientificreports/ Accepted online: 09 October 2023)
[3] Breaking The Silence: An innovation ASL to Text Conversion System Leveraging Computer Vision & Machine Learning for Enhanced Communication, Pooja Bagane, Muskan

Thawani, Prerna Singh, Raasha Ahmad, Rewaa Mital, Obsa Amenu, Jebessa (International Journal of Intelligent System and Application in Engineering, ISSN: 2147-6799, IJISAE, 2024, 12(14s), 246)

Text books:
[1] "Digital Image Processing, 3$^{rd}$Edition" by Rafael C. Gonzalez and Richard E. Woods (2008)
[2] "Pattern Recognition and Machine Learning" by Christopher M. Bishop (2006)
[3] "Computer Vision: Algorithms and Applications" by Richard Szeliski (2010)

Conference proceedings:
[1] Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
[2] Proceedings of the International Conference on Machine Learning (ICML)
[3] Proceedings of the International Conference on Computer Vision (ICCV)

Online journals with a DOI (Digital Object Identifier):
[1] Coversion of  Sign Language To Text, Akash Kamble, Jitendra Musale, Rahul Chalavade, Rahul Dalvi, Shrikar Shriyal (International Journal For Research in AppliedScience & Engineering Technology (DOI: https://doi.org/10.22214/ijraset.2023.51981).

Online journals without a DOI
[1] Real-Time Indian Sign Language Recognition Using Convolutional Neural Networks, Snehal Hon, Manpreet Sidhu, Sandesh Marathe, Tushar A. Rane, Volume 9, Issue 2 February 2024