Realtime Indian Sign Language Recognition and Sentence Generation

Anannya Banerjee 10200119057

Jeet Nath 10200119065

Anjali Sahani 10200119067

Sneha Mondal 10200119071

Abstract — **Communication is the most crucial thing for humans. Sign language is the only medium of communication for mute and deaf people with the rest of society. Various research has been done on this corresponding field in various countries of the world, but not so much has been done on Indian Sign Language. Indian Sign Language (ISL) is a gesture language that gives linguistic information using hands, pose and facial expressions. The aim of this project is to develop a system that can recognise a few expressions of Indian Sign Language in real time and generate continuous sentences from them. Here, we properly distinguish between flexible and fixed signs based on hand-based and pose-based features. In this project, the MediaPipe holistic API has been used to correctly detect and track hands and pose in the frame obtained from a normal web camera. After the feature extraction, the LSTM model is trained using sequential gesture data from 30 consecutive frames for every expression. The proposed model for real-time ISL recognition and sentence generation is cost-effective, computationally inexpensive, does not require any special equipment or gear, and outperforms a few previous works in this field.**

Keywords- **Sign Language, Gestures, Hands, Mediapipe, Indian sign language**

I. INTRODUCTION

A. Motivation and Background:

According to the 2011 census, there are nearly 2.68 crore (around 2.21% of the total population) deaf and dumb people in India. A marginal increase has been seen in this population, rising from 2.19 crore in 2001 to 2.68 crore in 2011 [1]. It is extremely disheartening to see the gap that this has created in Indian society. With the improvement of Human-Computer Interaction (HCI), more research in this field is helping computers act as mediators between signers and non-signers. With this project, we plan to narrow down the gap and help re-join their community with our society by introducing an inexpensive Indian Sign Language Recognition technique that will help non-signers understand the meaning of a particular sign without the help of any interpreter. So, the disabled will be able to communicate with us easily and we will be able to understand what they are trying to say.

B. Indian Sign Language:

Major efforts have been made by the Indian deaf community, various NGOs, and numerous organizations like All India Federation of the Deaf (AIFD), and the National Association of the Deaf to encourage Indian Sign Language (ISL) in India and finally, after 2001, Ali Yavar Jung National Institute of Hearing and the Handicapped (AYJNIHH) established India's first ISL cell in Mumbai. In 2005, the National Curricular Framework (NCF) approved sign language as an optional third language choice for hearing students. In 2006, NCERT published a chapter on sign language in a class III text book, emphasizing that sign languages are also just another mode of communication. ISL differs in many aspects from other countries' sign languages. ISL uses hand gestures, facial expressions, body postures, the location of the hand with respect to the body and a lot more to represent signs.

C. Challenges:

For a few reasons, Indian Sign Language is more complicated than other sign languages.

1) Despite the fact that ISLRTC has been established to standardize ISL, ISL is still in its early stages.
2) As said in the previous section, ISL needs hand gestures, facial expressions, body postures and a lot more. So, it makes ISL too complicated as most of the signs need both hands together.
3) Complicated hand shapes
4) For some signs, the hand makes contact with a specific part of the body.
5) The hand's position in relation to the body contributes to the signs.
6) Sometimes two signs look similar, like the signs for the alphabet "V" and the number "2".
7) Some of the expressions combine multiple signs.

II. LITERATURE SURVEY

1. Introduction

There are various techniques available which can be used for the recognition of sign language. Different authors have used different techniques according to the nature of sign language and the signs considered. A lot of work has been done on static signs, but unfortunately, till date, not much research work has been reported for dynamic signs in Indian Sign Language. Different researchers use numerous types of approaches to recognize sign language. We will discuss some of these approaches.

2. Previous Works

i. Glove based

- Rinalduzzi [2] used a Magnetic Positioning System using 7 magnetic nodes to track the position and orientation of 5 fingers and used a technique called MagIK after mounting all the nodes on a glove to derive all the gestures. Then an SVM classifier is used to train between different characters.
- J. Weissmann [3] used Cybergloves, which measured the angle of 18 hand joints. As features, they used the angle made by neighbouring fingers, wrist pitch, and thumb rotation and later used ANN to classify between different characters.
- M.A. Mohandes [4] proposed a method for the recognition of two-handed Arabic signs using the Cyber Glove and support vector machine. The Principal Component Analysis (PCA) feature is used for feature extraction. This method consists of 20 samples of 100 signed by one signer. 15 samples of each sign were used for training the Support Vector Machine to perform the recognition. The system was tested on the remaining five samples of each sign. A recognition rate of 99.6% on the testing data was obtained.

ii. Vision based

- In the work of Mathavan Suresh Anand [5], they are following vision-based approach. After getting the image, they are following three steps for the recognition to happen. First the hand gets separated from the image by segmentation using the Otsu algorithm, then the feature-extraction is done by the DWT algorithm, and for the classification they use KNN classifiers.
- The team of Joyeeta Singha [6] is using K-L algorithm. After skin-detection, they are using K-L Transform algorithm for the feature extraction. For the classification, they used two methods: one is the angle made by the eigen vector with the reference axis, and the other one is finding the Euclidian distance between the eigen vectors of the test image and the eigen vectors of the database image.
- Lekhashri and Pratap [7] developed a system for both static and dynamic ISL gesture recognition. The various features extracted are skin tone areas, temporal tracing, and spatial filter velocimetry. This obtains the motion print of the image sequence. Then pattern matching is used to match the obtained motion prints with the training set, which contains the motion prints of the trained image sequences. Then, the closest match is produced as the output.
- According to the work of Geetha M [8], after capturing and pre-processing the image, they are using 4 more steps before feature extraction, which are tracing the hand boundary, finding MCPs in that boundary, B-Spline approximation to generate a Bezier curve, and finally smoothing out the Bezier curve. For the feature extraction, they are using a unique way of dividing the curve area into 8 regions and finding the KMCP points. Finally, SVM is used for the classification.
- The team of Jungpil Shin [9] used three different datasets for ASL. Those datasets contained a total of 780000, 1815, and 65774 images of ASL signs, respectively. For feature extraction, they are using Mediapipe and getting 21 3D landmark co-ordinates and calculating every possible distance between these 21 3D landmarks and angles between these different points. Then for classification, they are using SVM and Light GBM. For the first dataset, they are getting an accuracy of 87.60%, and for the latter two, they are 99.39 and 98.49%, respectively.

III. THEORITICAL BACKGROUND

1. Mediapipe Holistic hand and pose detection:

Holistic by MediaPipe (by Google) helps in real-time detection of human poses, face landmarks, and hand tracking. Separate models for poses, faces, and hands are available from Mediapipe. But they are all optimised in their own particular domain and because of their own specializations, input to one component is not suited for the other one. So Google introduced Mediapipe Holistic to integrate all components into one, which treats the different regions of the image with component-appropriate specifications. So "MediaPipe Hands" is one of the components of the Mediapipe holistic, which is a high-resolution hand and finger persuit answer and "MediaPipe Pose" is an ML solution for high-fidelity body pose tracking with 33 inferred 3D landmarks and also a background segmentation mask on the whole body from RGB video frames.

Mediapipe Hands [10] wants to infer 21 3D landmarks of a hand from one frame. MediaPipe Hands makes use of a machine learning pipeline that consists of many models that work together: a palm detection model that uses the whole image to come up with an aligned hand bounding box; a hand landmark model that returns hi-fi 3D hand keypoints from that cropped image space known by the palm detector with the correct inputs to the MediaPipe solutions API; Fig. 3-1 gives a visible representation of the hand landmarks.

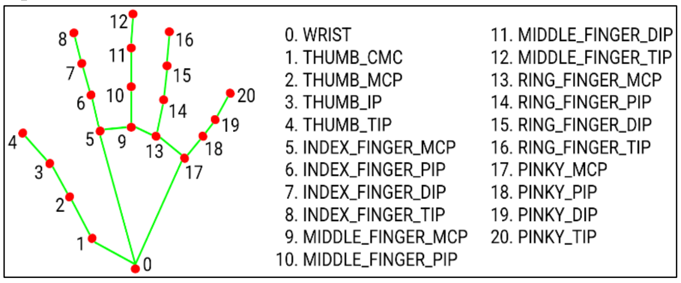


Fig 3-1: Mediapipe Hands Landmarks

From one frame, Mediapipe Pose tries to infer 33 3D landmarks of the body. This also makes use of an ML pipeline that consists of mainly two models: a person/pose detection model (BlazePose Detector) that detects the person; and a pose landmark model (BlazePose GHUM 3D) that renders 33 3D pose landmarks. Fig. 3-2 gives a visible representation of the pose landmarks.

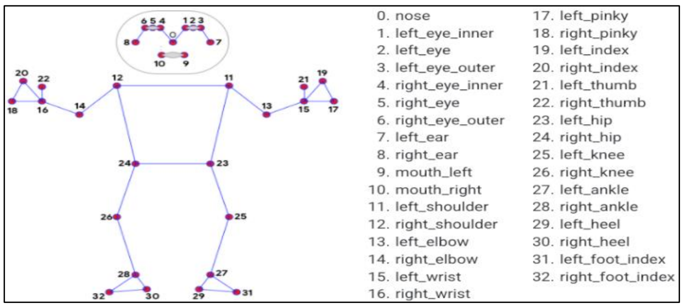


Fig 3-2: Mediapipe Pose Landmarks

LEFT_HAND_LANDMARKS:
This returns a list of 21 hand landmarks on the left hand. Each landmark consists of x, y, and z and is normalized to [0.0, 1.0] by the image width and height, respectively. z represents the landmark depth, with the depth at the wrist being the origin. The smaller the value of z, the closer the hand is to the camera.

RIGHT_HAND_LANDMARKS:
This returns a list of 21 hand landmarks in the right hanin the same representation as right_hand_landmarks.

POSE_LANDMARKS:
This returns a list of 33 pose landmarks. Each landmark consists of x, y and z and is normalized to [0.0, 1.0] by the imagewidth and height, respectively. z represents the landmarks depth with the depth at the midpoint of hips being the origin.

2. 3D Translation and Euclidian Distance in 3D plane:
3D Translation:

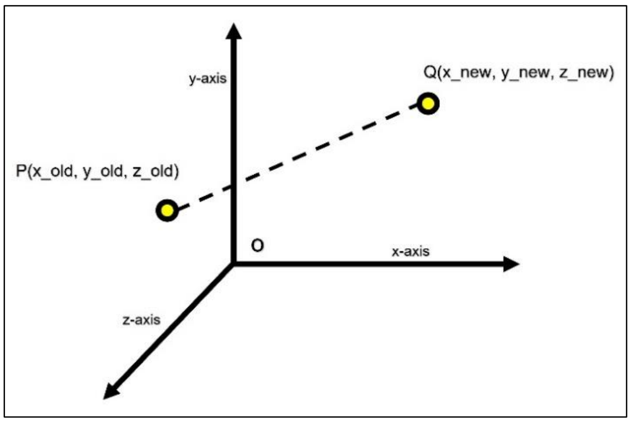


Fig 3-3: Translation in a 3D plane

As Fig 3-3 shows, suppose the initial co-ordinates of a point P is (x_old, y_old, z_old) and the after translation the new co-ordinate is (x_new, y_new, z_new). Now if the translation vector or shift vector is (T_x, T_y, T_z) then,

$$\begin{bmatrix} \boldsymbol{x\_new} \\ \boldsymbol{y\_new} \\ \boldsymbol{z\_new} \\ \mathbf{1} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} & \boldsymbol{Tx} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \boldsymbol{Ty} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \boldsymbol{Tz} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \text{X} \begin{bmatrix} \boldsymbol{x\_old} \\ \boldsymbol{y\_old} \\ \boldsymbol{z\_old} \\ \mathbf{1} \end{bmatrix}$$

OR

$$\mathbf{x\_new = x\_old + T_x}$$
$$\mathbf{y\_new = y\_old + T_y}$$
$$\mathbf{z\_new = z\_old + T_z}$$

Euclidian Distance in 3D plane:

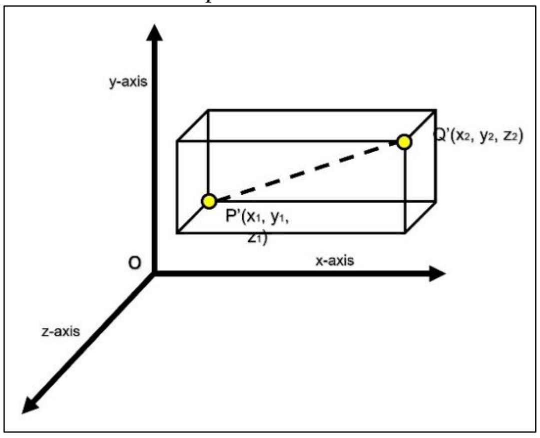


Fig 3-4: Euclidian Distance in 3D plane

As shown in Fig 3-4, consider two points P' (x_1, y_1, z_1) and Q' (x_2, y_2, z_2), then the Euclidian distance between these two points is,

$$\mathbf{d\ (P', Q') = \sqrt{\ [(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2]}}$$

3. Angle between two lines in 2D plane:

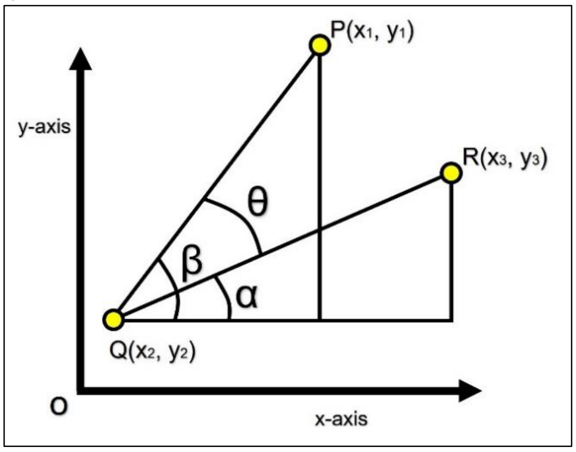


Fig 3-5: Angle between two lines in 2D plane

As the Fig 3-5 shows, there are three points P (x_1, y_1), Q (x_2, y_2) and R (x_3, y_3) which make an angle 'θ' on Q. Here $\theta = \beta - \alpha$. So, $\tan\theta = \pm \tan(\beta - \alpha) = (\tan\beta - \tan\alpha) / (1 - \tan\beta \tan\alpha)$. So, θ can be derived by,

$$\boldsymbol{\theta = \tan^{-1}((\tan\beta - \tan\alpha) / (1 - \tan\beta\tan\alpha))}$$

[where, $\boldsymbol{\tan\beta = ((y_2 - y_1) / (x_2 - x_1))}$ and $\boldsymbol{\tan\alpha = ((y_3 - y_2) / (x_3 - x_2))}$]

IV. PROPOSED SYSTEM

1. Proposed Framework:

ISL is, as said in the introduction, quite difficult. Some of the signs just require one hand in the frame, while others require both. Based on this, we divided ISL expressions into two categories: SingleHand and DualHand. We divide it into two further sections after that. Some of them are "Flexible Signs," which means that the hands can be placed wherever in the frame. That sign's meaning will not be altered. For some of the signs, the distance from the body must be addressed, as those movements must be performed in a specified location of the body. "Fixed Signs" was the term we gave to these signs. Our classification of ISL is properly shown in the Fig 4-1 with example.

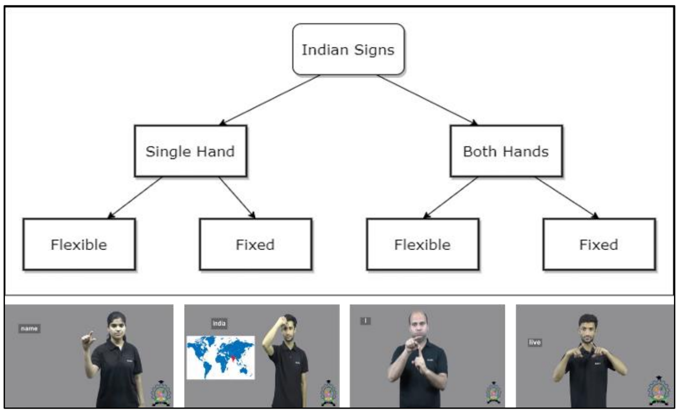


Fig 4-1: ISL classification

As the Fig 4-2 shows, there are a total of four sections in the proposed system: Imageset Collection, Feature Extraction, Training and Prediction. All are explained with detail in the next section.

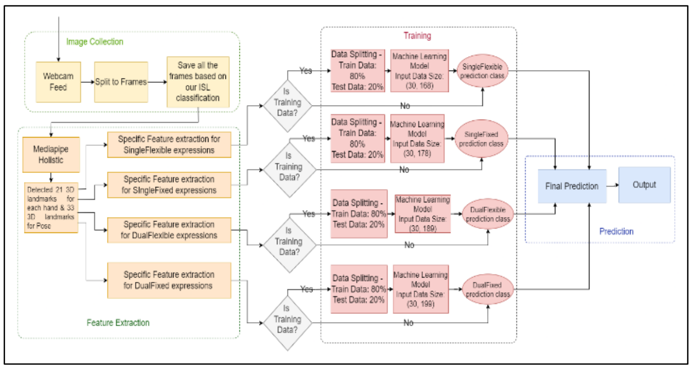


Image Collection
Webcam Feed
Split to Frames
Save all the frames based on our ISL classification
Mediapipe Holistic
Detected 21 3D landmarks for each hand & 33 3D landmarks for Pose
Specific Feature extraction for SingleFlexible expressions
Specific Feature extraction for SingleFixed expressions
Specific Feature extraction for DualFlexible expressions
Specific Feature extraction for DualFixed expressions
Feature Extraction
Is Training Data?
Yes
No
Training
Data Splitting - Train Data: 80% Test Data: 20%
Machine Learning Model Input Data Size: (30, 168)
SingleFlexible prediction class
Machine Learning Model Input Data Size: (30, 178)
SingleFixed prediction class
Machine Learning Model Input Data Size: (30, 189)
DualFlexible prediction class
Machine Learning Model Input Data Size: (30, 199)
DualFixed prediction class
Final Prediction
Output
Prediction

Fig 4-2: Proposed System for ISL Recognition

Stage 1: Imageset Collection

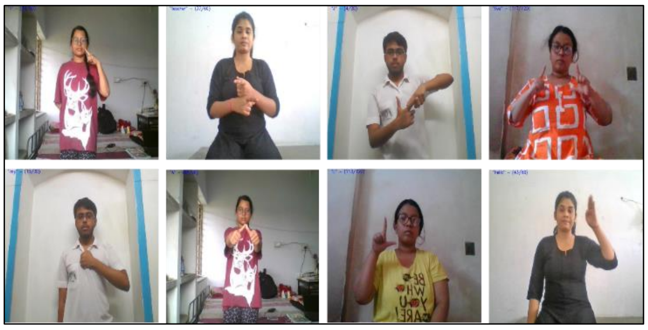


Fig 4-3: Gathered Imageset Sample

In this project, we tried to work with a total of 19 ISL expressions. Because of the unavailability of the ISL dataset online, the four of us captured the ISL imageset ourselves (Fig 4-3). For each expression, we gathered a total of 120 sequences and for every sequence, there are a total of 30 frames. All the frames are captured using our normal laptop webcam in 640 x 480 resolution. Based on our ISL classification, we stored every frame sequence of every expression.

The 19 ISL expressions are divided into 4 sections as follows:

SingleFlexible: L, name, this, jayga, --
SingleFixed: Hello, my, India, I_word
DualFlexible: A, B, I_lett, J, language, N, sign, teacher, W, --
DualFixed: Kol, Live

"--": It is a shift character as per the proposed system. Both Flexible ISL characters have one shift character in the list. The shift character is trained using the images of corresponding fixed characters. For example, the shift character of SingleFlexible is trained with 30 (as 120/4 = 30) sequences of each expression of SingleFixed characters and the shift character of DualFlexible is trained with 60 (as 120/2 = 60) sequences of each expression of DualFixed characters.

"jayga": A few of the expressions in ISL are a mixture of signs. For example,

- Grandfather = Sign of Man + Sign of Old
- Gray = Black + White + Mingle

Just like that, for a place, there is also a mixture of signs. At the end of every place sign, there is a gesture with no name. So, we named this gesture "jayga". Delhi is a mixture of "Del" and "jayga". As the Fig 4-4 shows, same goes for West Bengal. It is a mixture of the sign of "W", "B", and, of course, at the end, the sign of "jayga".

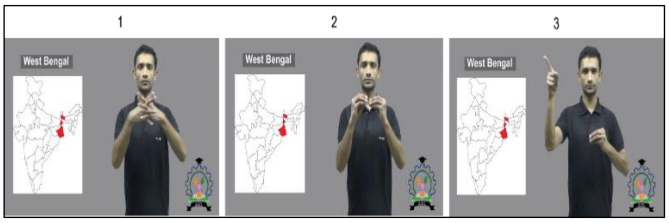


Fig 4-4: Sign Mixture for West-Bengal

Stage 2: Feature Extraction

Four different combinations of features are being extracted for the four different sections of ISL expressions. As for the flexible signs, only the hand features are enough; no pose features are extracted for them. And as the fixed signs are related to special positions as per the body, we have to consider the pose features too for them.
For Flexible Signs only hand features are extracted.

SingleFlexible: (The features are depicted graphically in Fig 4-5)

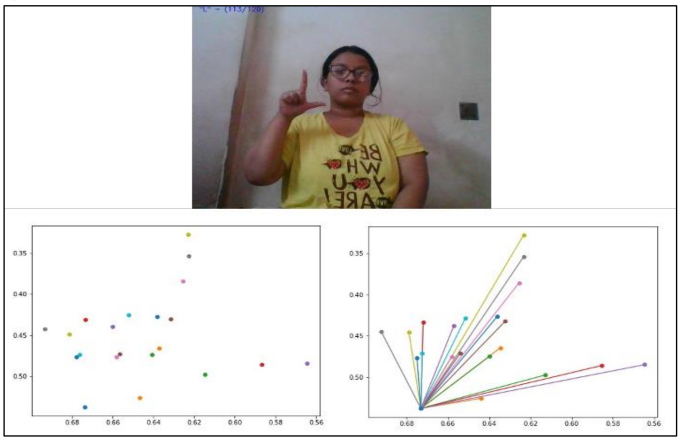


Fig 4-5: Feature Extraction for SingleFlexible

- 21 3D translated co-ordinates of the hand, so that wherever we keep our hand in the frame, it won't matter much.
- 3D euclidian distances from the wrist to the other 20 hand co-ordinates.

The size of the feature array of a single sequence of SingleFlexible expression is (30, 168).

DualFlexible:(The features are depicted graphically in Fig 4-6)

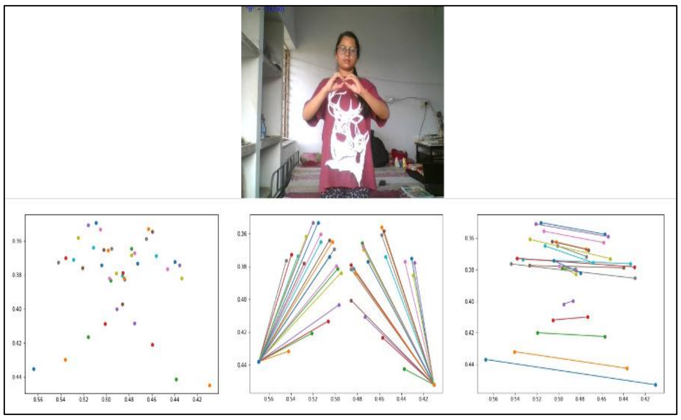


Fig 4-6: Feature Extraction for DualFlexible

- 21 3D translated co-ordinates of the hand, so that wherever we keep our hand in the frame, it won't matter much.
- 3D euclidian distances from the wrist to the other 20 hand co-ordinates.
- Distances in 3D Euclidian space between the corresponding landmark co-ordinates of both hands.

The size of the feature array of a single sequence of SingleFixed expression is (30, 178).

For fixed signs, we have to consider the pose features too.

SingleFixed: (The features are depicted graphically in Fig 4-7)

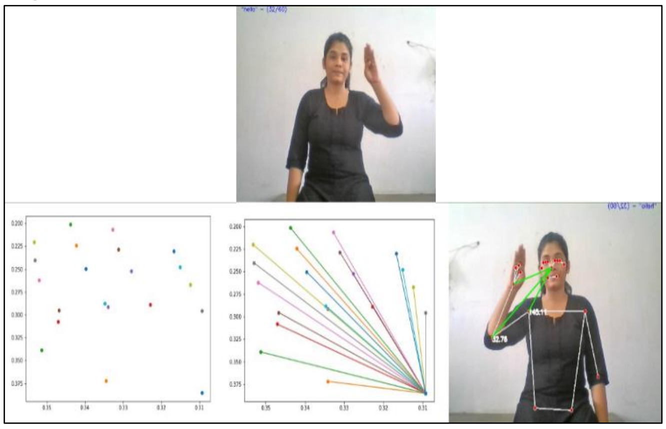


Fig 4-7: Feature Extraction for SingleFixed

- 21 3D translated co-ordinates of the hand.
- 3D euclidian distances from the wrist to the other 20 hand co-ordinates.
- The angle at the elbow and shoulder of the visible hand.
- We took the nose co-ordinate as our pivot co-ordinate to calculate the distance from. So 3D Euclidian Distance from the nose to the wrist-elbow-shoulder of the visible hand.

The size of the feature array of a single sequence of DualFlexible expression is (30, 189).

DualFixed: (The features are depicted graphically in Fig 4-8)

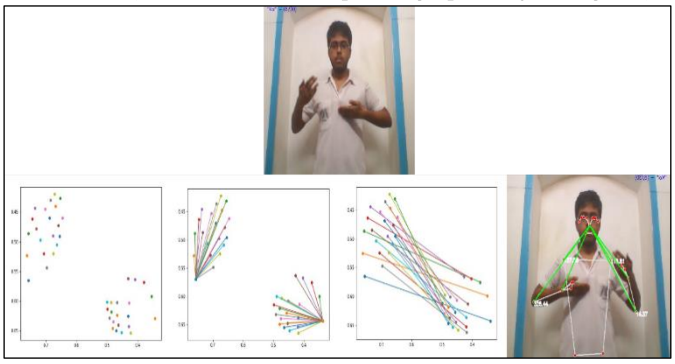


Fig 4-8: Feature Extraction for DualFixed

- 21 3D translated co-ordinates of the hand.
- 3D euclidian distances from the wrist to the other 20 hand co-ordinates.
- Distances in 3D Euclidian space between the corresponding landmark co-ordinates of both hands.
- The angle at the elbow and shoulder of the visible hand.
- We took the nose co-ordinate as our pivot co-ordinate to calculate the distance from. So, 3D Euclidian Distance from the nose to the wrist-elbow-shoulder of the visible hand.

The size of the feature array of a single sequence of DualFixed expression is (30, 199).

Stage 3: Training

We made four separate classification models for the four sections of ISL expressions. The model summary for all four of them is completely the same except for the input size, which depends on the feature array. Here is the list of algorithms we used in our machine learning model:

LSTM: Long Short-Term Memory networks–usually just called "LSTMs"–are a special kind of RNN, capable of learning long-term dependencies. They work tremendously well on a large variety of problems and are now widely used. LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior; it's not something they struggle to learn. LSTMs perform significantly better in the sequence-to-sequence framework and attention-based learning [11]. As ISL is full of gestures, we are using a sequence of 30 frames for each expression. For this, LSTM is the best algorithm to use.

Dense: In any neural network, a dense layer is a layer that is deeply connected with its preceding layer, which means the neurons of the layer are connected to every neuron of its preceding layer. This is also known as a fully connected layer, and is a layer that is used in the final stages of the neural network. This layer helps the model establish the relationship between the values of the data set on which the model is working by modifying the dimensionality of the output from the previous layer.

Activation Functions: An activation function defines or restricts the output of a node given an input or set of inputs. Basically, an activation function decides what and whether to send any value to the next node. Here, two types of activation functions are used:

1) **ReLU:** ReLU or Rectified Linear Unit, is a simple activation function that is often used. The main advantages of ReLU over other functions are that it does not activate all nodes at once, and its linearity after (> 0) allows for simple optimization. ReLU's function is f(x) = max (0, x).
2) **Softmax:** Softmax converts a vector of numbers into a vector of probabilities. Naturally, the Softmax activation function is used in the output layer of a multi-class classification model like this one.

Optimizer: Here, the ADAM optimizer is used. Adaptive Moment Estimation, or ADAM, is an algorithm used for optimization technique for gradient descent. This method is straightforward to implement, computationally efficient, has little memory requirements, and is well suited for problems that are large in data or parameters [12].

Loss Function: A loss function is used to evaluate the model by computing the distance between the current output of the algorithm and the expected output. As this is a multi-class classification model, we used categorical-crossentropy as the loss function.

Stage 4: Prediction and Quantitative Analysis

Quantitative Analysis:

To analysis the performance of these four models, we used classification metrics such as accuracy and confusion matrix.
Accuracy: The accuracy of the model (ACC_{ISLR}) is defined as the proportion of correctly classified predictions to the total number of predictions,

$$ACC_{ISLR} = [(TP + TN) / (TP + FP + FN + TN)]$$

[TP: True Positive, TN: True Negative, FP: False Positive, FN: False Negative]

Confusion Matrix: Confusion Matrix is a tabular visualization of the ground-truth labels versus model predictions. Each row of the confusion matrix represents the instances in a predicted class, and each column represents the instances in an actual class.
The Train Loss Curve, Train Accuracy Curve, and Confusion Matrix of these four models are shown below with their accuracy value:

SingleFlexible: (0.95)

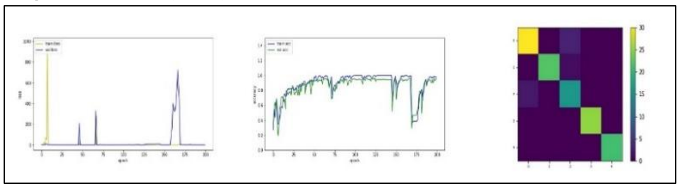


Fig 4-9: Performance Analysis of SingleFlexible

SingleFixed: (0.989583)

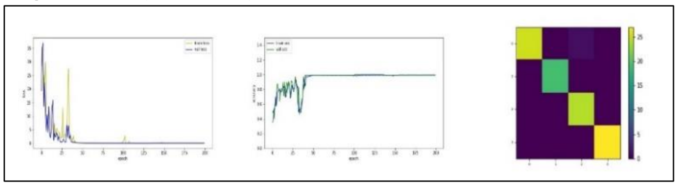


Fig 4-10: Performance Analysis of SingleFixed

DualFlexible: (0.991667)

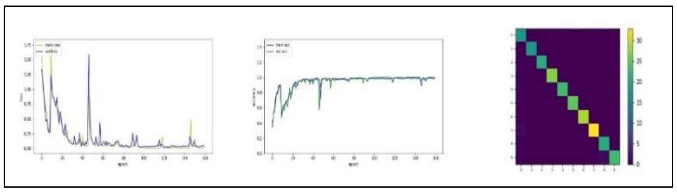


Fig 4-11: Performance Analysis of DualFlexible

DualFixed: (0.958334)

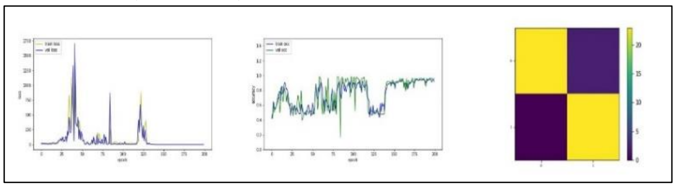


Fig 4-12: Performance Analysis of DualFixed

Prediction:

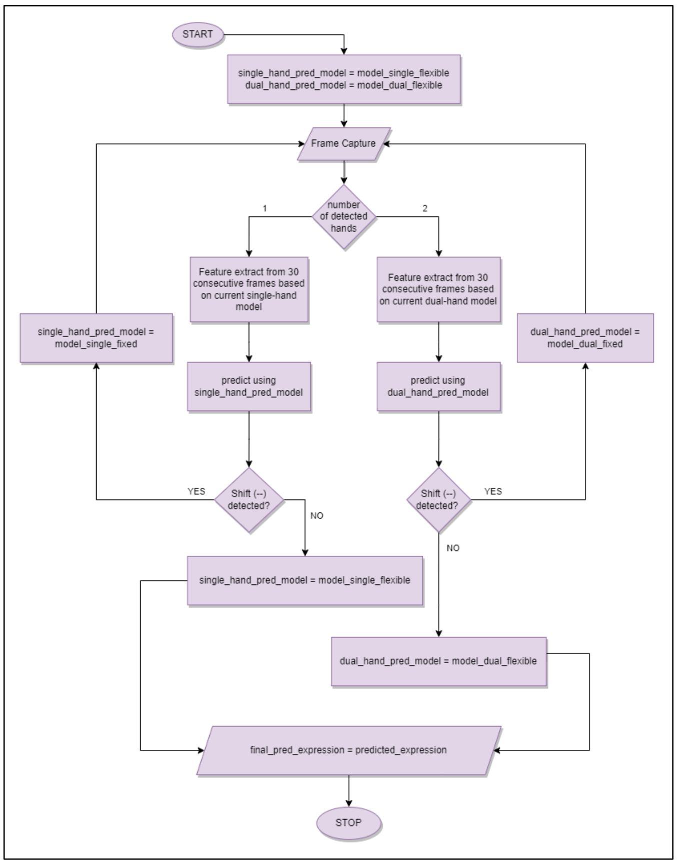
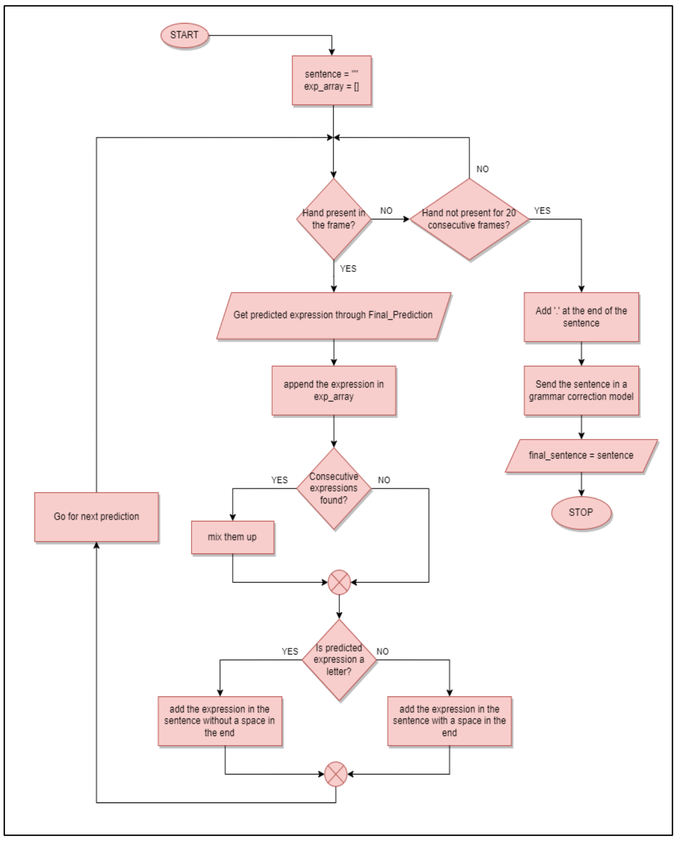

START
single_hand_pred_model = model_single_flexible
dual_hand_pred_model = model_dual_flexible
Frame Capture
number of detected hands
1
2
Feature extract from 30 consecutive frames based on current single-hand model
Feature extract from 30 consecutive frames based on current dual-hand model
single_hand_pred_model = model_single_fixed
dual_hand_pred_model = model_dual_fixed
predict using single_hand_pred_model
predict using dual_hand_pred_model
YES
Shift (--) detected?
NO
Shift (--) detected?
YES
NO
single_hand_pred_model = model_single_flexible
dual_hand_pred_model = model_dual_flexible
final_pred_expression = predicted_expression
STOP

Fig 4-13: Flowchart diagram of Prediction

The flowchart in Fig. 4-13 clearly depicts the final prediction workflow of our project. As all the models are made separately and the final prediction is happening in real time for every expression of every section, we have to utilize four of our models perfectly. As the flowchart describes perfectly, by default our Single Hand Prediction Model is SingleFlexible and Dual Hand Prediction Model is DualFlexible. If the shown expression belongs to SingleFlexible or DualFlexible section, it will predict that properly and move on to the next prediction. But if the expression belongs to the Fixed category (SingleFixed or DualFixed), then by the SingleFlexible or DualFlexible model, it will be predicted as a Shift Character (--) and for the next iteration, the Single Hand Prediction Model will be shifted to SingleFixed, or in case of two hands, the Dual Hand Prediction Model will be shifted to DualFixed. Then the fixed frame features will be extracted from the next 30 frame sequences, and after the prediction, the model will again shift back to the default prediction model for the next prediction.

2. Sentence Generation:



START
sentence = ""
exp_array = []
NO
Hand present in the frame?
NO
Hand not present for 20 consecutive frames?
YES
YES
Get predicted expression through Final_Prediction
Add '.' at the end of the sentence
append the expression in exp_array
Send the sentence in a grammar correction model
final_sentence = sentence
YES
Consecutive expressions found?
NO
STOP
Go for next prediction
mix them up
YES
Is predicted expression a letter?
NO
add the expression in the sentence without a space in the end
add the expression in the sentence with a space in the end

Fig 4-14: Flowchart Diagram of Sentence Generation

The sentence generation after the prediction is properly shown in the Fig 4-14 flowchart. The predicted expression is appended to an array. After every prediction, we look for the sequence of mixed signs of ISL, like "W," "B," and "jayga" for "West Bengal" in the said array. If found, they are mixed and replaced inside the array. All the expressions in the array are appended to a sentence variable. To understand the end of a sentence, we are checking the availability of hands in the frame. It is depicted that the sentence is over if the hand is not available for 20 frames in a row.array. All the expressions in the array are appended to a sentence variable. To understand the end of a sentence, we are checking the availability of hands in the frame. It is depicted that the sentence is over if the hand is not available for 20 frames in a row.

A transformer is an ML model that adopts the mechanism of self-attention, differently weighting the significance of each part of the input data. It is primarily used in Natural Language Processing (NLP) and computer vision. Here a transformer model named Gramformer has been used. Gramformer is a popular grammar correction model. So, to make the sentences grammatically correct, we are using this pre trained model which most of the time gives accurate result.

V. EXPERIMENTAL RESULTS AND DISCUSSION

Experimental Results:

Setup 1: The videos of the signs of ISL were captured using a normal laptop webcam. The distance from the camera should be such that the head, both arms and torso can be seen properly. For setup 1, we asked a few people to use the project in real time. Fig 5-1 shows that all the signs are recognized successfully. There was quite a variety in the background, available light, camera angle. But our model worked perfectly in every situation.

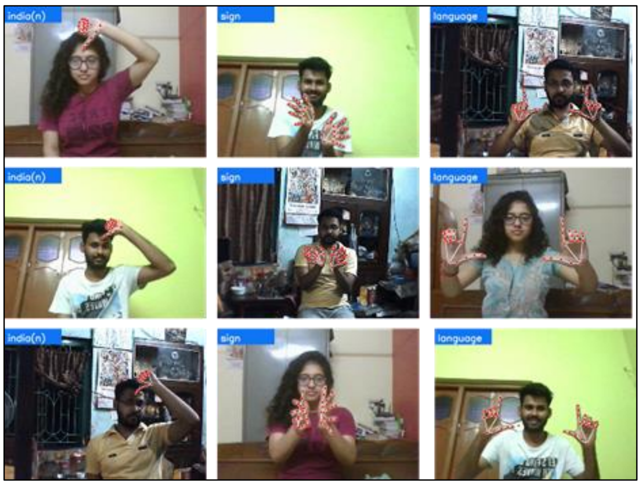


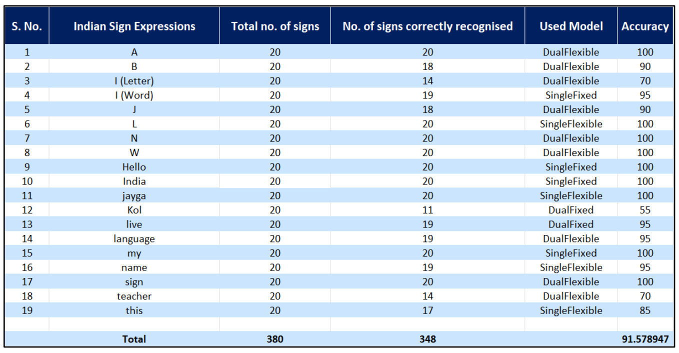
Fig 5-1: Project Testing

Setup 2: For setup 2, we asked five different people to capture images of the 19 ISL expressions. We captured a total of 20 frame sequences (4 from each signer) for 19 different ISL expressions. After processing, the system accuracy result was generated.

Results and Discussion:

The imageset for the testing is gathered from multiple signers who are completely different from the people used for training. The five signers we asked included users whose ages ranged from 18 to 65 years. We gathered 4 videos from each person, for a total of 20 videos were tested for each expression. As Fig 5-2 and Fig 5-3 shows the accuracy of this recognition system is 91.578947%. We evaluate the performance of the entire Indian Sign Language Recognition system in terms of accuracy. The accuracy is calculated as follows:

$$Accuracy = \frac{correctly\ classified\ gestures}{total\ number\ of\ gestures}$$



S. No.	Indian Sign Expressions	Total no. of signs	No. of signs correctly recognised	Used Model	Accuracy
1	A	20	20	DualFlexible	100
2	B	20	18	DualFlexible	90
3	I (Letter)	20	14	DualFlexible	70
4	I (Word)	20	19	SingleFixed	95
5	J	20	18	DualFlexible	90
6	L	20	20	SingleFlexible	100
7	N	20	20	DualFlexible	100
8	W	20	20	DualFlexible	100
9	Hello	20	20	SingleFixed	100
10	India	20	20	SingleFixed	100
11	jayga	20	20	SingleFlexible	100
12	Kol	20	11	DualFixed	55
13	live	20	19	DualFixed	95
14	language	20	19	DualFlexible	95
15	my	20	20	SingleFixed	100
16	name	20	19	SingleFlexible	95
17	sign	20	20	DualFlexible	100
18	teacher	20	14	DualFlexible	70
19	this	20	17	SingleFlexible	85
	Total	**380**	**348**		**91.578947**

Fig 5-2: Classification results of the proposed approach

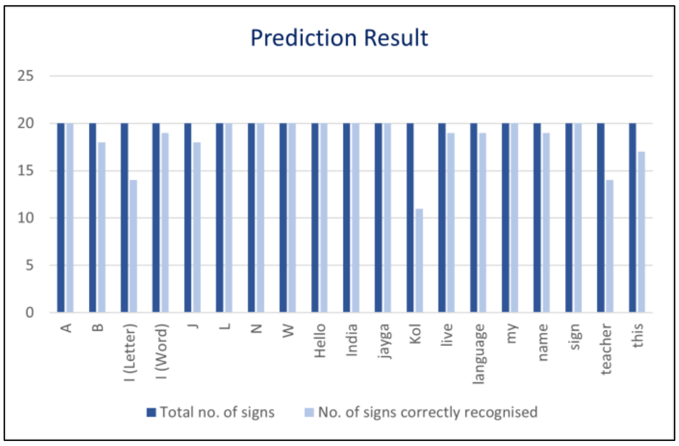


Fig 5-3: Graph showing total signs and correctly classified signs

VI. CONCLUSION AND FUTURE WORK

Advantages:

- There is no need for any special equipment or gear just to recognize signs.
- There is no special region or some specific position in the frame where it detects signs.
- Our project has a success rate of nearly 91.57%.
- Our project not only recognizes Indian signs correctly but also generates near perfect sentences.

Limitations:

- Mediapipe detection gets wobbly if the background gets complex, there is some complicated hand position, the lighting in the room is not apt, or if the hand color gets nearly similar to the background color.
- Even if we are getting a good accuracy rate while training, we are unable to get the same amount of accuracy in real world testing.
- Right now, two similar expressions or characters one after another, can't be predicted separately and added into the sentence. Ex. JEET or ANANNYA... Here 'double E' or 'double N' can't be predicted consecutively.
- As we are making sentences from the detected expression, we have to predict the correct expression and we are waiting some time for the prediction to stabilize. So, to detect an expression, our final prediction takes about 7-8 seconds, which is quite a lot.

Future Works:

- Gather more datasets from different people.
- Use image augmentation, image transformation, and image scaling for better real-life prediction.
- Try to decrease the final prediction time so that the prediction can happen more quickly.
- Find a way to predict two of the same expressions consecutively.
- ISL expressions include not only hand gestures but also facial expressions and poses. This project is done using only the hand and pose. Working on facial expressions is still in progress.
- Without using shift characters for model shifting, we can use model calibration algorithms like temperature scaling to calibrate the model to give a more realistic confidence score, and model shift can happen based on that confidence score.

VII. REFERENCES

1. Persons with Disabilities (Divyangjan) in India - A Statistical Profile: 2021, Government of India, Ministry of Statistics and Programme Implementation, National Statistical Office, Social Statistics Division, www.mospi.gov.in

2. Matteo Rinalduzzi, Alessio De Angelis, Francesco Santoni, Emanuele Buchicchio, Antonio Moschitta, Paolo Carbone, Paolo Bellitti and Mauro Serpelloni, "Gesture Recognition of Sign Language Alphabet Using a Magnetic Positioning System", Appl. Sci., 11, 5594, 2021

3. J. Weissmann and R. Salomon, "Gesture Recognition for Virtual Reality Applications Using Data Gloves and Neural Networks", IEEE, 1999, pp. 2043-2046

4. M. A. Mohandes," Recognition of Two-handed Arabic Signs using the CyberGlove," The Fourth International Conference on Advanced Engineering Computing and Applications in Sciences,2010

5. Mathavan Suresh Anand, Nagarajan Mohan Kumar, Angappan Kumaresan, "An Efficient Framework for Indian Sign Language Recognition Using Wavelet Transform", Circuits and Systems, 2016, 7, 1874-1883

6. Joyeeta Singha , Karen Das, "Hand Gesture Recognition Based on Karhunen-Loeve Transform", Mobile & Embedded Technology International Conference 2013

7. B. Lekhashri and A. ArunPratap,"Use of motion-print in sign language recognition," IEEE National Conference on Innovations in Emerging Technology (NCOIET), pp. 99-102, 2011.

8. Geetha M, Manjusha U C, "A Vision Based Recognition of Indian Sign Language Alphabets and Numerals Using B-Spline Approximation", International Journal on Computer Science and Engineering (IJCSE), Vol. 4 No. 03, ISSN : 0975-3397

9. Jungpil Shin, Akitaka Matsuoka, Md. Al Mehedi Hasan and Azmain Yakin Srizon, "American Sign Language Alphabet Recognition by Extracting Feature from Hand Pose Estimation", Sensors 2021, 21, 5856

10. Zhang, F.; Bazarevsky, V.; Vakunov, A.; Tkachenka, A.; Sung, G.; Chang, C.L.; Grundmann, M. Mediapipe hands: On-device real-time hand tracking. arXiv 2020, arXiv:2006.10214.

11. Ralf C. Staudemeyer, Eric Rothstein Morris, "– Understanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks", arXiv:1909.09586v1 [cs.NE] 12 Sep 2019

12. Diederik P. Kingma; Jimmy Lei Ba. ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION, arXiv:1412.6980v9 [cs.LG] 30 Jan 2017