International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

Analysis of Vision based Techniques for the Translation of Indian Sign Language

[1]Attili SubhaVidisha ,[2]Anubuthi Kottapalli, [3]Anshula Aithal, [4]Amanpreet Kaur Pawa, [5]Ashwini M Joshi
[1,2,3,4,5]Dept. Of CSE PES University
[1]subhavidisha@gmail.com,[2] anubuthi.kottapalli@gmail.com, [3]anshula.aithal02@gmail.com, [4]amanpreetpawa2@gmail.com, [5]ashwinimjoshi@pes.edu

Abstract— Sign language acts as a medium of communication among those of the hearing impaired and mute community. However, it cannot be easily understood by common people. Various research has been done to bridge this gap by developing Sign Language Recognition (SLR) methodologies. Studies say that 1 in every 5 deaf people is Indian.

In this paper, a thorough review of these methodologies has been done, to compare and contrast various aspects of them. This includes an overview on different preprocessing methods used like segmentation, image morphological processing, cropping, etc, feature extraction techniques like Fourier Descriptors, Image Moments, Eigen values, Mediapipe and others. This study also covered classification models spanning from Distance metrics to Kernel based approaches and feedforward neural networks, along with Deep Learning based methods such as CNNs, LSTMs, GANs, Transformers etc.

Index Terms—Computer Vision, Deep Learning, Feature Extraction, Morphological Image Processing, Mediapipe.

I. INTRODUCTION

Hearing disabilities affect a big portion of the world. Despite this, different nations have adopted different sign languages to communicate with each other. Some of these recognized sign languages include American Sign Language, British Sign Language, Indian Sign Language, Turkish Sign Language, Chinese Sign Language, etc. The WHO has estimated approximately 63 million Indians to be hearing disabled [1]. This indicates that an average of 6.3% of the population suffers from such auditory impairment. Despite the number of hearing-impaired people being high, the number of certified translators and interpreters for the same is drastically low, with approximately only a few low hundred interpreters across India[2]. Hence, this solution acts as a bottleneck when considering the population of the hearing-disabled community. With the advent of Machine Learning and Computer Vision, various methods have been researched to make Sign Language Recognition and Translation more ubiquitous to the aurally impaired. However, different sign languages have different semantics and intricacies to them. For example, the American Sign Language uses only 1 hand to gesture alphabets whereas the Indian Sign Language uses both hands for most of its alphabets. These types of differences brought about a variety of methods suitable for studying various kinds of features in various forms. In this paper, we aim to provide a comprehensive summary of all the methods explored in creating a functional real-time Indian Sign Language Recognition model, along with their benefits and limitations.

II. INDIAN SIGN LANGUAGE RECOGNITION SYSTEM

Most sign languages are not simply a sequence of hand movements. Rather, they involve a combination of hand movements with reference to their body frame and facial expressions.

The Indian Sign Language is one that consists of both static and dynamic gestures. Static symbols include those where a single motion of the user's hands is enough to indicate the word. Examples of these kinds of gestures include those done for numbers and alphabets. Dynamic gestures include those that require performing a sequence of actions to depict a certain phrase. These can also incorporate facial expressions at times, to express the intensity or emotion behind an action. Examples of these kinds of symbols include day-to-day sentences normally spoken by everyone.

A. Hardware Based Methods

Hardware-based methods rely on the use of devices to record the motion performed by the user. A typical example is the use of glove-based methods, where the signer wears a pair of gloves fitted with sensors to capture his/her motion.

Glove-based methods using flex sensors were the most popular choices for sign language translation[3][4]. These methods typically make use of flex sensors and contact sensors that are supplied with a certain voltage which changes with the movement of fingers because of the associated change in resistance. The advantage of methods of this kind is that they can record even those intricate gestures performed that are normally missed by the human eye. Most sign languages comprise thousands of movements, some more subtle than others. Incorporating sensors to capture each one of these movements increases the system's robustness and accuracy. However, tracking the user's hand motions requires him to carry heavy equipment, making it quite problematic. These gloves can also be quite expensive, making them inconvenient.

B. Computer Vision Based Methods

Vision based methods involve recording the action performed by the signer and classifying it as a certain gesture using some suitable classification method. Unlike sensor-based methods, these methods do not require the user to be fitted with any kind of sensors, making it a more convenient method. However, as the actions are being classified via a video input, less intricate details pertaining to the action may not be recorded, depending on the equipment used. Occlusion of certain parts of one hand by the other may also occur, resulting in loss of relevant information.

To enhance the performance of such a model, various steps are included in vision-based systems to capture and extract even the minute features from the image input. The more relevant features captured, the more robust the model is. However, one must be careful and take steps to ensure that the model does not learn any unnecessary features as this may misclassify the intended action.
The following steps elucidate the most common methods generally practiced to ensure correctness of the model:

Data Acquisition: The first step in any model for sign language recognition is selecting a suitable source of data. Vision based models would typically use a dataset of images or videos. The model/system would learn from this data and is expected to classify actions of similar type. Image databases befit static symbols, whereas video databases are normally employed to store both static and dynamic gestures. The resolution of the images/videos, along with their dimensions play a significant role in the ease of computation and accuracy achieved while classifying them. Frames with a significantly high resolution capture more details, increasing the acuity of the image captured. However, this in turn can increase the space and file sizes required to the same when preprocessing and extracting relevant features.

Large scale, authentic datasets for the Indian Sign Language are not readily available. Hence, most researchers choose to create their dataset by themselves.

P. C. Badhe et al [5] collected videos of gestures performed by the members of the deaf and mute community of "Ali Yavar Jung National Institute for Hearing Handicapped, Bandra, Mumbai". This data was primarily composed of alphabets of the English language, digits and phrases. The phrases included were "India", "Hello", "Namaste", "Thank You", "Sorry", "Please", "Fire", "Danger", "Help Me" and "I Am Hungry". The entire dataset included a total of 130,000 videos, split between the testing and training phases.

R. Patil et al [6] collected static symbols of digits of ISL as images, recorded from their webcamera.

J. Bora et al [7] used a dataset of 9 Assamese alphabets, including images of subjects performing gestures in the Assamese Sign Language. These images were used by Microsoft Kinect, which enables retrieval of the depth of each pixel, making 3D hand estimation more accurate and an RGB webcamera, that recorded 2D images.

S. C J et al [8] used a dataset consisting of the hand silhouettes of the binary image of the signers performing static ISL gestures. The original dataset was collected from Rahmaniya HSS Special School Calicut. It comprises of 2500 images, performed by 7 different signers. Images from this dataset were introduced with random translations and scaling factors to create an augmented dataset, comprising 5157 images.

J. Singha et al [9] constructed a dataset consisting of 24 alphabets in Indian Sign Language, with each letter having 10 samples, resulting in a dataset consisting of 240 images.

J. Rekha et al [10] constructed their own database. 10 images of each of 23 letters were recorded for their training dataset, and 20 videos each of the other 3 letters{h,j,y} were recorded to create their dynamic video dataset of the same.
Y. Rokade et al [11] utilised a dataset publicly available for ISL alphabets and discarded the symbols for letters such as {C, I, L, M, N, R, U}, possibly to avoid hand-over-hand actions from impairing their model. The images consisted only of the hands, and the background kept black across all images.

R. Kumar et al [12] utilised a publicly accessible dataset comprising of 26 letters signed in the American Sign Language. Each letter or class consists of 4500 images of the same.

D. P. Papastratis.I et al [13] made use of 3 datasets to study Continuous Sign Language Recognition using glosses, i.e., association of a word with a sign. These datasets include RWTH-Phoenix-Weather-2014 dataset, CSL dataset (Chinese Sign Language) and GSL dataset (Greek Sign Language). The RWTH-Phoenix-Weather-2014 consists of videos of weather forecasts performed by 9 different signers. It spans over a vocabulary of 1295 words/glosses and contains 5672, 540, and 629 videos for training, validation, and testing, respectively. The CSL dataset is a publicly available one too, with 100 predefined sentences performed five times by 50 signers, amounting to 25,000 videos in total The GSL dataset comprises of five distinct and often encountered scenarios in various public services performed by seven different signers. It has 10,295 videos and 310 distinct glosses in its vocabulary. Each signer repeats signs from pre-defined discussions five times.
In [14] Bantupalli et al constructed a dataset of 100 signs from ASL, each sign was performed five times by a single signer in different lighting condition and varying speeds, but with the same background. They record videos using iPhone 6 on 60fps and 720p resolution. The authors also resized and rotated the images at random and augmented it to the dataset. The dataset for each sign consisted of 2400 images.

In [15] Aditya Das et al used an existing dataset that contains images of static signs representing 24 alphabets (A to Y excluding J) in American Sign Language. There was an average of 100 images per alphabet class. The authors used methods such as cropping, scaling and rotation to augment the data to reflect variations present in real world data.
Reshna S et al [16] created their dataset using their laptop camera. The dataset consists of 500 images of 11 symbols in the ISL system: Gestures for A, B, C, I, L, Namaste, O, U, V, W and Hi.

Kaushal Goyal et al [17] used webcam with Mediapipe Holistic and opencv to build their dataset. Their dataset consisted of 26 static signs representing the alphabets and 5 words/phrases (“hello”, “thanks”, “I love you”,”Indian”,”how are you doing”) for gesture-based signs. For each sign they took 60 videos with 30 frames per video. Each frame has 1662 key points that were stored in an array.

B. Subramanian et al [18] used a real-time dataset that includes 30 videos for each of 13 sign gestures, captured at 640 × 480 resolution under different conditions. The model's performance was also tested on two benchmark datasets: WLASL, which

features various signing styles, lighting and backgrounds, and LSA64, containing 3200 videos of 64 isolated sign gestures.
S. Alyami et al [19] utilized two datasets: KArSL and LSA64. KArSL is an Arabic Sign Language dataset consisting of 100 dynamic signs. These are performed by 3 signers and each sign is performed 50 times by each signer. The LSA64 contains 64 signs in Argentinian Sign Language. These are performed by 10 signers and each sign is performed 5 times. 30 frames are extracted from each video.

K. Shenoy et al [20] used a dataset containing 33 gestures. The dataset includes 0-9 in Indian Sign Language. Each digit has an average of 1450 Images. The dataset also consists of alphabets in Indian Sign Language except 'h', 'j' and 'v'. There are an average of 300 images per alphabet. captured. There are 9 gesture-related intermediate hand poses such as “Thumbs Up”, “Sun Up”, about 500 images per pose were captured. The dataset totally contains 24,624 images. For training the HMM, 15 videos are captured for the 12 single handed gestures. All the images and videos are captured with various resolutions.
P. Likhar et al [21] created their own dataset, which was captured using an RGB camera utilising BlazePose, Mediapipe Hands and BlazeFace. They captured a total of 1045 videos of 55 sentences, using varying temporal speeds and camera positions. The dataset was divided into 15:2:2 for training, validation and testing.

K. Tripathi et al [22] made use of an external camera to capture their dataset. They used a black background and concentrated only on the upper body. They captured 10 sentences 10 times each using 5 different people and divided into 6:4 for training and testing.

N. Keshari et al [23] used the dataset from Microsoft Research which consists of 1000 classes covering around 25000 videos for train, validation and test datasets.

In [24] they review multiple papers where data is acquired from vision-based methods and glove-based methods. For vision-based methods most are capture using cameras resulting in RGB images.

A.Halder et al [25] dataset consists of 156000 ASL alphabets, 4972 ISL alphabets, 12856 Italian Sign Language Alphabets, 1400 ASL numbers and 4124 Turkey Sign Language numbers.

G. H. Samaan et al [26] generated their own dataset called DSL10-Dataset which consists of 750 videos divided randomly for training and testing. All videos were recorded in indoor environments with neutral lighting and an average camera at 30 frames per second.

G. Rao et al [27] created a dataset consisting of 18 words. The dataset also considered a phrase 'No Sign' when no hands are detected within the frame.

D. Kothadia et al [28] also created their own dataset. It is comprised of 11 words, with 1100 videos for each word. The videos were performed by both male and female participants and were captured under normal conditions, i.e., normal brightness and orientation, with a neutral background. The average length of the videos was 2s.

M. Kumar [29] used a dataset consisting of 26 signs. Each sign is gestured 10 times. The same images are training and testing.
In [30], the authors made use of the RWTH-Phoenix-Weather-2014T dataset, which is a corpus of large vocabulary comprising 1066 gestures and 2887 German words.

Preprocessing Data: Preprocessing methods are employed to ensure that the models do not learn unnecessary information and noise from the input image/frame. This step involves taking raw images as input and formatting them into a more suitable form, to extract maximum number of relevant features from it.

(i)Cropping: Cropping and resizing is a common measure used to ensure that all the frames fed to the system are of uniform dimensions. Often, the cropped image includes only the important object in consideration. In the case of sign language recognition, this would be the hands and physiognomy of the signer [5][9]. Often, the images are resized to ensure uniformity in sizes when passed to an image-based neural network such as CNN.[8].
In [14] the authors trimmed the videos to only include gestures which extended up till the neck. The authors of [16] resized the image to new dimensions and stored them into an array, these images are passed through a Sobel filter for detecting the edges (vertical and horizontal). Similarly, the author of [15] cropped the images to only include the area that is recognized and remove any background information. In [23] they shifted the width and height of the RGB frames by a 0.2 scale and used a bounding box to crop the videos to get rid of unwanted space and text present in the videos.

(ii)Image Segmentation: It is another method commonly utilised to partition the input image into distinct image regions, based on pixel values and a predetermined threshold value [6]. Segmented images are easier to handle as they represent only the important attributes of the image. This simplifies the process of extracting features from them. Erosion, Dilation and Skin color detection are common preprocessing steps done to convert raw imagerial data into a more processable form [5][11][6]. Erosion and Dilation are morphological transformations that make use of a kernel (structural feature) which decides the nature of the output image after applying this kernel [6].

P. C. Badhe et al [5] incorporated a skin detection technique using a YCbCr model to continuously detect motion of the hands based on changing skin area.

Y. Rokade et al [11] also employed skin detection as a preprocessing measure using adaptive probabilistic models. An RGB-adjusted form of the input image was fed to this model. It converted the detected skin area of the frame into a binary image, whose skin value was determined and was subsequently normalised. Further noise and errors from the binary image were eliminated using Erosion and Dilation.

Rachana Patil et al [6] utilised similar segmentation techniques such as Context Subtracting, Skin Color Detection, and Edge Detection along with Erosion, Dilation and Blurring to remove noise and improve the quality of the frames fed to the model.
In [16] the authors used background subtraction algorithm to remove unwanted objects from the background and skin segmentation to segment hands and head of signers.They used

YCbCr colour space for skin detection in skin segmentation. The authors convert the RGB colour space to YCbCr using available formulas. A histogram is computed for all three components and the threshold value is determined, the images are spatially filtered for skin pixels and then the threshold is applied, and the image is smoothened. The output image only has skin pixels of the hand.

J. Singha et al [9] implemented a skin filtering algorithm that involves converting the RGB image to a HSV image. They further apply filters and apply smoothing techniques for background elimination, and ultimately concentrate on the largest connected region of skin-colored pixels, referred to as the Biggest Binary Linked Object (BLOB). Further, the hand is cropped by considering the minimum and maximum positions of the white pixels in the BLOB.

S. C J [8] detected the face of the signers in the binary image using Viola-Jones Face detection algorithm and the detected face was replaced with black pixels. Following this, a skin-segmentation algorithm, along with a largest connected component algorithm was employed to detect the region of the hands for further classification.

J. Rekha et al [10] preprocessed the frames from their database by performing skin segmentation in YCbCr color space. RGB based images are converted to YCbCr images using the equations:

$$Y = 0.299R + 0.587G + 0.114B \quad (1)$$

$$Cr = R - Y \quad (2)$$

$$Cb = B - Y \quad (3)$$

The input image frame colour vector is compared to the stored skin colour model data, and the pixels with the shortest distance are considered to represent skin. Those skin pixels are returned to RGB space. Hand is identified in each frame by taking the skin region threshold value into account.

K. Shenoy et al [20] performed face elimination by Histogram of Oriented Gradients followed by a linear SVM. Once it detects a face, the face contour is identified, and the entire face-neck region is blackened. The image is then converted from RGB to YUV. It identifies weather a certain region is skin by applying the following conditions:

$$80 < U < 130$$
$$136 < V < 200$$
$$R > 80 \,\&\, G > 30 \,\&\, B > 15$$
$$|R - G| > 15$$

They further apply morphological operations to remove noise. In this case they have used erosion and dilation. Hand stabilization is achieved using facial reference, utilizing the Kernelized Correlation Filter. In order to track the hand, the centroid of the hand is calculated in every frame. If there was movement, then the centroid would have changed. The slope between the centroid of the hand in the previous and current frame helps in determining the direction of motion.

In [20] they extracted the foreground image from the complete image thus getting the skeleton of the upper body. Further, hand region was subtracted by eliminating largest connected region thus obtaining only the hands as the result.

G. Rao et al [27] capture videos using the front camera at 30 frames per second. They apply a 3-fold 2D Gaussian Filter to the frames to reduce sharp variations and smoothen the frames. They then calculate the 2D gradient of the frame using the 1D gradient in both the x and y directions. Edge adaptive thresholding is applied where the block variational mean of each 3x3 Sobel mask is used as a threshold. Finally, they get the hand and head contour by applying a differential morphological gradient and finding the connected component.

M. Kumar [29] uses the Otsu algorithm to segment the hand from the background. He then applies morphological filters like erosion and dilation to remove noise.

(iii) Mediapipe Hands Detection: The release of open-source frameworks such as Mediapipe by Google has expedited the preprocessing phase in Computer Vision tasks. It utilises an inbuilt palm detection and hand landmarks detection model to annotate 21 unique landmark points on each hand presented in the frame.

R. Kumar et al [12] mentioned Mediapipe to enable hand tracking in real-time with their model. This involved annotating and storing 21 locations on each hand.

J. Bora et al [7] also used Mediapipe and collected the x,y and z coordinates of 21 landmarks from the left and right hands. The z coordinate was discarded for every image, as only the x and y coordinate were necessary for training their model.

B. Subramanian et al [18] utilized Mediapipe Holistics and Blazepose to collect face, hands and pose landmarks. The corresponding landmarks are split into Region of Interests for task-specific models. Overall, 1662 landmark points per frame are generated.

S. Alyami et al [19] used Mediapipe Pose Estimator to extract 75 key points from each frame; 21 from each palm, 11 from the face and the remaining from arms, hands, shoulders, legs and feet. Each key point consists of the x, y and z coordinate. However, they discarded the z coordinate, since the resultant model was simpler and performed better. 53 key points from palms and face were used the train the model.

A. Halder et al [25] used Mediapipe to extract x,y and z coordinates of 21 points from each hand and stored them for further processing after discarding the z coordinate. In [21] they made use of BlazePose, Mediapipe and BlazeFace while capturing their dataset.

Feature Extraction: Feature Extraction is a crucial step in image processing tasks. It entails capturing patterns and aspects from imagerial data and transforming them into numerical representations. As the number of such patterns may be quite high, compression techniques are used to reduce the dimensionality of the same.\\ Common dimensionality reduction techniques include Principle Component Analysis (PCA), Fourier Descriptors, Distance Transformations, Central and Hu Moments etc.

(i)Fourier Descriptors: Fast Fourier Transforms or FFTs are typically used in signal and image processing to decompose the signal/image into its sine and cosine component respectively. A Fourier Descriptor's fundamental idea is to use the Fourier transformed border, that is obtained by coefficients of a Fourier

function, as the shape feature. Image moment is a particular weighted average of pixel intensities of the image, or a function of such moments [32]. Computer Vision tasks typically use FFTs for extracting the high information geometric structure of the image.

P. C. Badhe et al [5] used 28x28 Fourier descriptors to extract features for further steps.

Y. Rokade et al [11] extracted features by performing distance transformation on their input image using Euclidean distance, denoting the distance of every pixel with its closest boundary pixel. Further Fourier Descriptors were applied on the projections of this distance transformed image, following which feature vectors were created using Central and Hu moments[11].

R. Patil et al [6] mentioned features such as form, color features, histograms and other geometrical features such as position, angle, distance, etc which were recorded and further compressed to reduce the dimensionality of the processed image.

(ii)Eigenvalues & Eigenvectors: J. Singha et al [9] extracted the Eigen vector and Eigen values from the input image with dimension 50 by 50. Only 5 out of the 50 Eigen vectors were considered.

(iii) Principle Curvature Based Region Detector and 2-D Wavelet Packet Decomposition: J. Rekha et al [10] performed feature extraction using 2 methods - Principle Curvature Based Region Detector and 2-D Wavelet Packet Decomposition. PCBR is a structure-based detector which depends on structural features of an image such as lines, edges, curves, etc. to differentiate between different regions of the image. These features tend to be robust to changes in intensity, colour and positions, and hence can be used by the detector to extract features from images that might represent the same objects, but vary in orientation, illumination and depth. 2-D Wavelet Packet Decomposition enabled extraction of textural information of the image, by using Gray Level Co-occurrence Matrix (GLCM). GLCM are used for characterising textural information. Features such as inertia, total energy, entropy, contrast, local homogeneity, cluster shade, cluster prominence, and information measure of correlation are extracted from the GCLMs[10]. In addition to shape and textural features, additional features pertaining to finger count was also accounted for, because most ISL letters required the use of 2 hands. This was done by calculating the Convexity Defects arising from checking the Convex Hull property with the contour of the hand.
K. Shenoy et al [20] used a grid-based fragmentation technique in which the hand sample is divided into M x N blocks. These blocks are used to populate a feature vector containing M x N feature values. The feature value is calculated as follows:

$$\text{FeatureValue} = \frac{\text{Area ofHand Contour}}{\text{Area of Fragment}} \quad (4)$$

Using Principle Component Analysis and t-distributed Stochastic Neighbour Embedding, the dimensionality of the feature vector is reduced.
K. Tripathi et al [22] used gradient and orientation histogram to extract appropriate features. The dimensions of the feature are extracted using PCA.

G. Rao et al [27] applied the Discrete Cosine Transform to extract energy information from the hand and head contours. The head component doesn't move very often, so the DCT of the head component is quite constant. PCA is applied to the DCT component, this ensures that only unique components of the feature matrix are retained.

M. Kumar[29] applied PCA and the principle components are utilized as the main features.

D. Kothadiya et al [28] used a multi-layer system architecture. InceptionResNetV2 was used to extract feature vectors from the video frames. Keypoints were obtained for each such frame from every video sequence and stacked together.

(iv)Mediapipe Features: Features from Mediapipe's Hand Landmark Detection model are generally landmarks of unique points on the palm. Hence, they typically require different extraction and processing methods as compared to images, which are conventionally used for Sign Language Recognition.
R. Kumar et al [12] extracted features more suited to the landmarks annotated by Mediapipe. The landmark coordinates were recalculated relative to the hand's centre by subtracting the centre point coordinates from each landmark
coordinate for each frame. To ensure consistency across all hand sizes, each landmark was scaled by a factor that was derived from the bounding box dimensions of the hand in that particular frame. The normalized landmarks were then flattened to form a 1D vector, which would act as their feature matrix for further classification.

J. Bora et al [7] mentioned in their study that all the landmarks were adjusted to reflect their relative position to the base of the wrist, and later normalised by dividing each of them by the largest absolute coordinate value[7]. These modified landmarks were saved in a CSV file for further classification.

Kaushal Goyal et al [17] used Mediapipe Holistics to collect data from the video of the sign performed. The pose is estimated within the frame by locating the head, shoulders, arms, hips and legs. Then the face and hand landmarks are estimated: 468 key points for face and 21 key points on each hand. All the key points are combined to produce a holistic representation of the signer that enables precise tracking of the signer. A total of 1662 key points were captured for face, hand and pose for each of the 30 frames of the videos.

S. Alyami et al [19] extracted key points from Mediapipe. If any key point is missing, it is replaced by a zero instead of null value in order to get equal length feature vectors. The dataset was also augmented by performing random rotation with a rotation parameter between 5 and -5. They also augmented the dataset by scaling, with a random scale parameter between 0 and 1.
B. Subramanian et al [18] flattened, concatenated and stored the landmark points into a file. All the null entries are removed from the data. Further, labels are created for each class by associating frames with their corresponding classes.

G. H. Samaan et al [26] used Mediapipe to get x,y and z coordinates of both hands getting a total of 126 keypoints, and using Mediapipe Pose Estimation extracts 33 keypoints in 3D space obtaining a total of 132 keypoints. Further, they obtained 468 keypoints for the face in 3D space giving a total of 1404 keypoints.

Classification: After the relevant features have been extracted, classification of the input, based on these features needs to be done.

(i)Distance Metrics: Earlier methods involved calculating the difference between the feature matrix of the given input and those of the actions present in the database. Measures such as Euclidean distance [5][11][9], Eigen value Weighted Euclidean distance [9], Manhattan distance, Mahalanobis distance, etc. are used to gauge the closeness between 2 data points. The lesser the difference/distance between the two, the more similarity between them is. Therefore, the symbol from the dataset with the least distance is considered to be the most approximate and accurate output.

The occlusion of certain parts of the hand by the other tends to misclassify actions as said earlier. P. C. Badhe [5] mentions the same with respect to certain letters such as {C, O} and {M, N, R}.

G. Rao et al [27] attempted classification using 3 different distance metrics: Euclidean distance, Normalized Euclidean distance, and Mahalanobis distance. Mahalanobis distance showed the best result with an average Word Matching Score (WMS) of 90.58% for the same training and testing dataset. They have also used an ANN with a varying number of hidden layers. The ANN outperforms all the classifiers with a Word Matching Score of up to 96.9%. The ANN also performs well when different signers are used whereas the Mahalanobis distance-based classifier fails to do so.

M. Kumar [29] used Linear discriminant Analysis (LDA) in order to classify the gestures. LDA has two phases: the training phase and the recognition phase. In the training phase, every gesture is represented as a column vector. The vectors are then normalized with respect to the average vector. The algorithm then finds out the Eigen vectors of the covariance matrix. The gesture space vector projections are obtained from the Eigen vector matrix, and they are then multiplied by the gesture vectors. In the recognition phase, the gesture space projections are normalized and then projected to gesture space. Finally, the Euclidean distance is calculated between the gesture space projection and all the gesture space projections. The projection with the least distance is the corresponding gesture and is converted to text and voice.

(ii)Kernel Based Approaches: J. Rekha et al [10] proposed the use of a multi-class SVM based classifier to recognise different letters of ISL in their dataset. This model was fed with the feature vectors of the images, obtained after performing PCBR, WPD and Finger counting using Convex Hull. Their approach achieved a classification and recognition rate of 86.3%. The researchers also noted that the accuracy of dynamic gestures decreased due to the more variable nature of their features.

Reshna S et al [16] also used SVM for classification of gestures. A multiclass SVM decomposes M-class problem into a series of binary classification problems. They used an SVM classifier with C=2.67 and gamma is 5.383. But for testing, the video recorded is preprocessed and Bitwise AND operation is done between features of training and test images to obtain a new image, then the cv2 library is used to compare the gestures with gestures in database and print the output.

A.Halder et al [25] compared 7 different models including SVM, KNN, Random Forest, Decision Tree, Naive Bayes, ANN, and MLP, and found that SVM outperformed all other models.

(iii)Neural Networks: Y. Rokade et al [11] perform a comparative study of classification models, using ANN and SVM with a polynomial kernel respectively. Both models were implemented with different numbers of features. As the number of features was increased in the feature vector (6,8 and 13), the accuracy of both the models rose, with ANN performing better than SVM in all the cases [11].

J. Bora et al [7] trained a feed-forward neural network with the input being normalised Mediapipe landmarks. This network consisted of dense and dropout layers to avoid overfitting, ReLu activation function to at the intermediate layers, and a SoftMax function at the output layer. The model was used for classifying both 2D as well as 3D images in real-time, achieving 99% accuracy in both domains.

N. Keshari et al [23] used the I3D RGB Inception architecture model. This model takes elements such as filters of the pooling layers from the 2D model and adds a 3rd dimension to them. They used the pretrained model training on 100 classes and expanded to 200 classes and added their own layers to the last 50 layers to train their own data.

(iv)Deep Neural Networks: With the advent of Deep Neural Networks, machine learning algorithms such Recurrent Neural Networks, Convolutional Neural Networks, Long Short-Term Memory-RNNs were employed in computer vision tasks for action identification and action recognition tasks.

R. Patil et al [6] used a CNN model to classify preprocessed frames of imagerial data depicted digits of ISL. The activation function used was ReLu, along with 2 different optimizers - SGD and ADAM. SGD optimizer performed better for their model and achieved 100% accuracy with their training data and 81% accuracy on their validation data.

G. H. Samaan et al [26] compared LSTM, Bi-LSTM and GRU models, and determined wether face keypoints were essential in translation. They found GRU to have the highest accuracy and that face keypoints did not make a big difference to the accuracy while translating.

R. Kumar et al [12] also used a CNN based architecture. However, rather than feeding imagerial data as input, 42 landmarks of both the hands, that were obtained by Mediapipe Hand Landmarks Detection were fed to the model. This architecture and usage shows a CNN model's ability to recognise patterns from non-visual data as well. The final output is a

probability distribution over all the letters from A-Z, and the class that has the maximum probability is presented as the output [12]. This model achieved a high accuracy of 100% across all classes, demonstrating the efficiency of Mediapipe with CNN.

Aditya Das et al [15] also used a CNN model -Inception v3 model for classification. They received high accuracy on the training data when the learning rate was set to 0.05. The loss function used is cross entropy. They obtained a validation accuracy of an average of 90% peeking at 98%.

The authors of [17] performed a comparative study of 2 models namely the CNN model and the LSTM model. The data was reshaped into a 4D array to be used as an input to the CNN model. They used a sparse categorical cross entropy loss function, and SGD for back propagation. The LSMT model is provided with the input of 30 timestamps and 1662 features. ReLU and sigmoid activation functions are used, followed by a Bulk normalisation layer and a dropout of 0.2. Adam optimizer , learning of 0.001 AND Cross entropy loss is used. The result showed that CNN model performed better for Static signs whereas the LSTM model worked better for the gestures.

Kshitij Bantupalli et al in [14] used an ensemble of two models CNN and LSTM. They used a CNN model called Inception to extract spatial features and then used LSTM to extract temporal features; using outputs from the SoftMax layer and pool layer of CNN.
The authors retrained the Inception model. The model displayed 99% accuracy on the training set. The output from the SoftMax layer and pool layer are collected and fed to the LSTM model. Both the models used cross-entropy cost function, ADAM optimizer. The LSTM model had a dropout of 0.3 to prevent overfitting. The results from each were compared and found that the outputs from the SoftMax layer gave higher accuracies.

D. P. Papastratis.I et al [13] developed a Generative Adversarial Network (GAN) for generating a sequence of glosses, that describes the gestures performed in the video. This GAN network comprises of a Generator and a Discriminator. A video encoder acts as the generator and a text-modeling network is used as the discriminator. The video encoder is fed with the input videos, from which it extracts spatio-temporal features and generates a gloss sequence. The text-modeling network models gloss level and sentence level text to differentiate between the true and predicted outcomes of the model. The generator competes with the discriminator by attempting to provide better accuracy of its predicted accuracy, whereas the discriminator competes with the generator by differentiating better between the true gloss sequences and the predicted sequences. The metric used to gauge the model's performance was Word Error Rate (WER). The RWTH-Phoenix-Weather-2014 dataset showed a WER of 23.7% on its validation data, CSL dataset demonstrated a WER od 2.1% and GSL showed 2.87% on its validation dataset.
S. C J et al [8] performed classification of static gestures using a CNN model. This CNN model consisted of an input layer, 6 hidden layers, a dropout and an output layer. ReLu and SoftMax activation functions are used before and at the output layers, to give a predictive probability for each class. 80% of the dataset was utilised in the training phase, and the remaining 20% was used during the testing phase. The accuracy of the training phase was 99.93% and an accuracy of 98.64% was observed for the validation phase.
B. Subramanian et al [18] proposed a MOPGRU model which improves the learning ability of GRU and the overall performance. It does so by calibrating the resultant of the update gate by the reset gate. This improves the learning process of the GRU gating unit, thereby accelerates the convergence rate, eliminates the gradient depletion problem, and improves the overall learning efficiency.
S. Alyami et al [19] proposed 3 models, LSTM, TCN and Transformer respectively. For the KArSL dataset, LSTM obtained an accuracy of 8% for signer-dependent mode and 62.9% for signer-independent mode, whereas Transformer obtained an accuracy of 99.74% for signer-dependent mode and 68.2% for signer-independent mode, while TCN obtained an accuracy of 88.9% for signer dependent mode and 59.8% for signer-independent mode. Applying the random rotations to augment the data increased the accuracy by 1%, whereas applying random scaling boosted the accuracy by 1.5%. However, on applying both the methods improved the overall accuracy by 5%. Including the face landmarks improved the Transformer performance by 4%. Transformer gives an accuracy of 91.01% for signer-independent mode and 98.25% for signer-dependent mode for the LSA64 dataset.

D. Kothadiya et al [28] implemented a multi-layer system consisting of a combination of LSTM and GRU to classify signs while capturing semantic dependencies. 6 different combinations of GRU and LSTM were considered: single-layer LSTM, double-layer LSTM (LSTM-LSTM), single-layer GRU, double-layer GRU (GRU-GRU), GRU followed by LSTM (GRU-LSTM), and LSTM followed by GRU (LSTM-GRU). The final output class is predicted by the output "SoftMax" layer. The LSTM-GRU model showed the highest accuracy from among the above-mentioned combinations. These models were also tested on other sign language datasets such as ASL, GSL, AUTSL (Ankara University Turkish Sign Language), in addition to the dataset which they created themselves. They achieved considerably high accuracies - 95.3%, 94.0% and 95.1% for the above used datasets and 97.1% on their own dataset.

III. CONCLUSION

After surveying multiple papers, we have drawn the following conclusions. Many researchers chose to construct their own dataset due to the lack of a publicly available dataset that was suitable for their study. These researchers mostly used their laptops/phone cameras to record gestures and images. Some authors used readily available datasets while excluding certain alphabets/gestures which were not compatible with their model. A few researchers augmented their dataset by rotating and resizing images to improve the robustness of the model.

We found that the most common preprocessing techniques implemented were cropping/resizing, where the dimensions of the frames were altered to ensure more uniformity, making it more appropriate for image-based models. Skin detection is a

popular image segmentation technique that was used to identify the hands from each frame using pixel values. Some researchers also used other image processing techniques, like Sobel mask and morphological transformations for object boundary detection.

The advent of Google's Open-Source Computer Vision Framework - Mediapipe has obviated the need for researchers to preprocess images in order to extract features. It has inbuilt models that provide key landmarks for pose, hand and face detection. These can be used as input features for various models.

Earlier methods for feature extraction include methods such as Fourier Descriptors, DCT, Eigen vectors and values, Central and Hu Moments, PCBR and WPD.

Various classification methods were used for the identification of gestures. Some of these include distance metrics which calculate the distance, i.e., the closeness between the appropriate features in the given input, and the same features in the database. Polynomial kernels such as SVMs were also applied to recognise different alphabets and gestures. Feed Forward Neural Networks were used to differentiate between gestures, by passing the feature vectors through a set of neural layers. The most commonly used Deep learning models include CNN and its variations and RNN models like LSTM. Both types of models are well suited to identifying imagerial data and sequential data, making them most suitable for sign language recognition. Other Deep learning models like GAN, Transformers and TCN are among the most recent advancements in this field. A summary of the same has been provided in Table I.

IV. FUTURE WORK

Many of the above studies have addressed the recognition of alphabets and digits of the Indian Sign Language. However, quite a few of the earlier studies excluded certain letters from their dataset due to the concealment of one hand by another while performing actions. This impediment was addressed by the introduction of Mediapipe which was able to estimate occluded hand parts as well.

Further, very few studies propose models for the recognition of complete ISL sentences, possibly due to a lack of such publicly available datasets for ISL sentences and words. However, recently, a few datasets for words have also been published. [32] and [33] are 2 such articles that have published large-scale ISL datasets. The former has around 4000 videos for different ISL-English words, while the latter has around 31,000 ISL-English sentence/phrase-pairs done. Further studies can be performed using such large-scale datasets, which improve the feasibility of building such models for the hearing disabled and mute community.

CNNs, LSTMs and other Deep Learning architectures can model the sequential dependencies across different parts of a video sequence for a given gesture/word. However, the semantic relations between multiple such words cannot be addressed by these same models to produce grammatically cohesive sentences. Additional architecture/methods must be employed to develop these relations between the identified words/phrases to form intelligible sentences.

TABLE 1
SIGN LANGUAGE RECOGNITION SYSTEMS

Title	**Methods**	**Remarks**
Indian Sign Language Translator Using Gesture Recognition Algorithm [5]	Skin Segmentation, Canny Edge Detection, Vector Quantization, Minimum Distance method	High accuracy. Certain alphabets were misclassified due to occlusions of certain hand parts by the other hand.
Indian Sign Language Recognition System [11]	Skin Segmentation, Distance transformation, Fourier Descriptors, Central and Hu moments, ANN, SVM	Only 17 letters of the English alphabet were used. ANN and SVM were compared, with ANN outperforming SVM.
Indian Sign Language Translation using Deep Learning [21]	Mediapipe Hands, Seq2Seq LSTM, Transformer	High accuracy observed for the transformer model in comparison to LSTM models.
Indian Sign Language Recognition using Convolutional Neural Network [6]	Contour detection, Bounding box, CNN, SGD, RMSprop	Only applicable to letters and digits, and not gestures in real time.
Continuous Indian Sign Language Gesture Recognition and Sentence Formation [22]	Skin color segmentation, Orientation Histogram, PCA, Distance Formulas	Limited dataset containing 18 words. Use of Sobel Filter makes it sensitive to lighting variations, motion blur, and camera vibrations.
Mediapipe and CNNs for Real-Time ASL Gesture Recognition [12]	Mediapipe, CNN	High accuracies observed for ASL alphabets. However, it is only applicable to ASL letters and not dynamic gestures.
Real-time Assamese Sign Language Recognition using MediaPipe and Deep Learning [7]	Mediapipe, Sequential Neural Network	High accuracy was observed. However, only 9 alphabets from the Assamese Sign Language were chosen for this study.

Real-time ASL to English text translation [23]	Bounding Box, I3D RGB Inception	Training a large dataset was computationally expensive.
Continuous Sign Language Recognition through a Context-Aware Generative Adversarial Network [13]	GAN, Video Encoder, Discriminator, gloss level and sentence level detection	This study made use of GAN to extract the most accurate gloss sequence from a given video. Also demonstrated SLR for dynamic gestures.
Real-time Vernacular Sign Language Recognition using MediaPipe and Machine Learning [25]	Mediapipe, SVM, KNN, Random Forest, Decision Trees, Naive Bayes, ANN, MLP	SVM was found to have the highest accuracy compared to all other models. They got 99 percent accuracy using SVM on a large dataset.
Mediapipe's Landmarks with RNN for Dynamic Sign Language Recognition [26]	Mediapipe, LSTM, bi-LSTM, GRU	They found face keypoints to not be contributing factor to overall accuracy.
Real-time Indian Sign Language (ISL) Recognition [20]	k-NN, HMM, Grid-based feature extraction	Limited dataset. The signer must wear a full-sleeved shirt. Lighting conditions must be ideal.
Isolated Arabic Sign Language recognition using a Transformer-based model and Landmark key points [19]	Mediapipe, LSTM, TCN, Transformer	High accuracy for signer-dependent mode whereas low accuracy for signer-independent model.
Indian Sign Language Recognition Using Eigen Value Weighted Euclidean Distance Based Classification Technique [9]	Euclidean distance, Eigen value weighted Euclidean distance	Small dataset and only limited to static signs.
An integrated Mediapipe-optimized GRU model for Indian sign language recognition [18]	Mediapipe, GRU	The authors considered a small dataset with only 13 gestures. Modified GRU improved model performance and learning rate.
American Sign Language Recognition using Deep Learning and Computer Vision [14]	CNN, LSTM	Accuracy dropped if a particular skin tone was not trained before or if the face of the signer was in the frame. The signer had to wear the same clothes throughout.
Sign Language Recognition Using Deep Learning on Custom Processed Static Gesture Images [15]	CNN Inception V3 model	The dataset considered was very small. They also used only one pretrained model, no other approaches were applied.
Conversion of Sign Language into Text [29]	LDA algorithm, Skin colour Segmentation	The results of the experiment were not presented.
Incorporating Relative Position Information in Transformer-Based Sign Language Recognition and Translation [30]	Bi-directional GRU, GRU-RSTs, Transformers	It does not work well for long sequences and non-overlapping signers.
Indian Sign Language recognition using Mediapipe holistic [17]	Mediapipe, CNN, LSTM	Considered a small dataset with very few dynamic gestures. Used Mediapipe to extract features making the model independent of lighting, skin color, and clothes worn.
Spotting and recognition of hand gestures for Indian sign language recognition system with Skin Segmentation and SVM [16]	Skin Colour Segmentation, SVM	The dataset considered was very small as it only included 11 static gestures, used skin segmentation, which can be affected by varying lighting conditions.

ACKNOWLEDGEMENTS

The authors would like to thank the Chairperson of the Department of Computer Science and Engineering, PES University for providing them with the opportunity and resources to conduct this research process.

REFERENCES

[1] S. Mishra, A. Jena, and P. Mishra, "Hearing impairment in India: A major public health challenge," The Lancet Global Health, vol. 10, no. 12, pp. e1668–e1669, 2022.

[2] I. J. of Applied Linguistics, "The status of sign language interpreting in India," Indian Journal of Applied Linguistics, vol. 1, no. 1, pp. 1–10, 2023.

[3] A. C. D. S. Sachin Bhat, Amruthesh M, "Translating Indian sign language to text and voice messages using flex sensors," 2015.

[4] M. Elmahgiubi, M. Ennajar, N. Drawil, and M. S. Elbuni, "Sign language translator and gesture recognition," in 2015 Global Summit on Computer Information Technology (GSCIT), pp. 1–6, 2015.

[5] P. C. Badhe and V. Kulkarni, "Indian sign language translator using gesture recognition algorithm," in 2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS), (Bhubaneswar, India), pp. 195–200, 2015.

[6] R. Patil, V. Patil, A. Bahuguna, and G. Datkhile, "Indian sign language recognition using convolutional neural network," ITM Web Conf.,vol. 40, p. 03004, 2021.

[7] J. Bora, S. Dehingia, A. Boruah, A. A. Chetia, and D. Gogoi, "Real-time Assamese sign language recognition using mediapipe and deep learning," Procedia Computer Science, vol. 218, pp. 1384–1393, 2023.International Conference on Machine Learning and Data Engineering.

[8] S. C J and A. Lijiya, "Signet: A deep learning based Indian sign language recognition system," pp. 0596–0600, 04 2019.

[9] J. Singha and K. Das, "Indian sign language recognition using eigenvalue weighted Euclidean distance based classification technique," International Journal of Advanced Computer Science and Applications, vol. 4, 2013.

[10] J. Rekha, J. Bhattacharya, and S. Majumder, "Shape, texture and local movement hand gesture features for Indian sign language recognition," in 3rd International Conference on Trends in Information Sciences Computing (TISC2011), pp. 30–35, 2011.

[11] Y. Rokade and P. Jadav, "Indian sign language recognition system," International Journal of Engineering and Technology, vol. 10, no. 21817, pp. 189–196, 2017.

[12] R. Kumar, A. Bajpai, and A. Sinha, "Mediapipe and cnns for real-time asl gesture recognition," 2023.

[13] D. P. Papastratis. I, Dimitropoulos. K, "Continuous sign language recognition through a context-aware generative adversarial network sensors,"2021.

[14] K. Bantupalli and Y. Xie, "American sign language recognition using deep learning and computer vision," in 2018 IEEE International Conference on Big Data (Big Data), pp. 4896–4899, 2018.

[15] A. Das, S. Gawde, K. Suratwala, and D. Kalbande, "Sign language recognition using deep learning on custom processed static gesture images," in 2018 International Conference on Smart City and Emerging Technology (ICSCET), pp. 1–6, 2018.

[16] S. Reshna and M. Jayaraju, "Spotting and recognition of hand gesture for Indian sign language recognition system with skin segmentation and svm," in 2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), pp. 386–390, 2017.

[17] D. V. G. Kaushal Goyal, "Indian sign language recognition using mediapipe holistic," 2023.

[18] B. Subramanian, B. Olimov, S. Naik, S. Kim, K.-H. Park, and J. Kim, "An integrated mediapipe-optimized gru model for Indian sign language recognition," Scientific Reports, vol. 12, p. 11964, 07 2022.

[19] S. Alyami, H. Luqman, and M. Hammoudeh, "Isolated arabic sign language recognition using a transformer-based model and landmark keypoints," ACM Trans. Asian Low-Resour. Lang. Inf. Process., feb 2023.

[20] K. Shenoy, T. Dastane, V. Rao, and D. Vyavaharkar, "Real-time Indian sign language (isl) recognition," in 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1–9, 2018.

[21] P. Likhar and G. Rathna, "Indian sign language translation using deep learning," in 2021 IEEE 9th Region 10 Humanitarian Technology Conference (R10-HTC), pp. 1–4, IEEE, 2021.

[22] K. Tripathi and N. B. G. Nandi, "Continuous Indian sign language gesture recognition and sentence formation," Procedia Computer Science, vol. 54, pp. 523–531, 2015.

[23] N. Keshari and A. K. Srikantham, "Real-time asl to english text translation," 2021.

[24] M. Safeel, T. Sukumar, K. Shashank, M. Arman, R. Shashidhar, and S. Puneeth, "Sign language recognition techniques-a review," in 2020 IEEE International Conference for Innovation in Technology (INOCON), pp. 1–9, IEEE, 2020.

[25] A. Halder and A. Tayade, "Real-time vernacular sign language recognition using mediapipe and machine learning," Journal homepage: www.ijrpr. com ISSN, vol. 2582, p. 7421, 2021.

[26] G. H. Samaan, A. R. Wadie, A. K. Attia, A. M. Asaad, A. E. Kamel, S. O. Slim, M. S. Abdallah, and Y.-I. Cho, "Mediapipe's landmarks with rnn for dynamic sign language recognition," Electronics, vol. 11, no. 19, p. 3228, 2022.

[27] G. Rao and P. V. V. Kishore, "Selfie video based continuous Indian sign language recognition system," Ain Shams Engineering Journal, 2017.

[28] D. Kothadiya, C. Bhatt, K. Sapariya, K. Patel, A.-B. Gil-Gonz´alez, and J. M. Corchado, "Deepsign: Sign language detection and recognition using deep learning," Electronics, vol. 11, no. 11, 2022.

[29] M. Kumar, "Conversion of sign language into text," 2018.

[30] N. Aloysius, M. Geetha, and P. Nedungadi, "Incorporating relative position information in transformer-based sign language recognition and translation," IEEE Access, vol. 9, pp. 145929–145942, 2021

[31] D. Shanmugapriya, "Fourier descriptors and moments." University Lecture, 2021.

[32] A. Sridhar, R. G. Ganesan, P. Kumar, and M. Khapra, "Include: A large scale dataset for Indian sign language recognition," in Proceedings of the 28th ACM International Conference on Multimedia, MM '20, (New York, NY, USA), p. 1366–1375, Association for Computing Machinery,2020.

[33] A. Joshi, S. Agrawal, and A. Modi, "Isltranslate: Dataset for translating Indian sign language," 2023 position information in transformer-based sign language recognition and translation," IEEE Access, vol. 9, pp.145929–145942, 2021.