Dual Mode Sign Language Communication Module - A Web Based Prediction Model

Mr. Adithya R[1], Mr. Karthik Nagaraj Naik[2], Mr. Prakhyath KS[3], Ms. Sampreeti S Harikant[4]

[1234] *Student, Department of Computer Science, Sahyadri College of Engineering and Management, Mangalore, Karnataka, India - 575007*

E–mail: *adithyaiyer72@gmail.com, karthiknaikwork2024@gmail.com, prakhyath2502@gmail.com, sampreetihalkar@gmail.com*

Abstract

Effective communication plays an important role in our daily lives, fostering interaction, idea exchange, and emotional expression. However, individuals who are mute or deaf encounter significant challenges when communicating with those who are not. Previous studies have predominantly focused on one-way communication, such as converting speech to gesture, using static datasets limited to alphabets, digits, and specific languages. This project aims to address the limitations of current systems by creating an app that enables two-way communication between individuals with hearing impairments and those without. It will utilize dynamic gestures and a centralized platform that can be accessed by everyone. The intended system comprises two modules: one for translating sign gestures into speech and the other for converting speech into 2D animated sign gestures. To achieve this, a dataset comprising of 10 words was recorded. Leveraging MediaPipe Holistic and OpenCV alongside a webcam, we recorded video sequences depicting individuals executing sign language gestures. Sign gesture-to-speech conversion is realized using a LSTM deep learning algorithm, while text-to-2D sign gesture conversion involves keyword extraction from live speech and subsequent development of 2D animated gestures based on these extracted keywords. Through this innovative approach, the objective of this system is to facilitate communication between people with hearing impairments and the broader community, aiming to close the communication divide.

Key words. American sign language, LSTM, text-to-2D sign algorithm, MediaPipe Holistic, Gesture recognition, Face recognition, SLR, SLT, OpenCV, Python

1. INTRODUCTION

Individuals who experience deafness or hearing impairment face daily challenges, particularly in communication. As per projections by the WHO (World Health Organization) [1], the global population affected by these conditions continued to rise significantly by 2030 and 2050, with India, the world's most populous country, housing approximately 63 million individuals with hearing impairments [1]. Despite India's strides in technological advancement and infrastructure development, addressing the requirements of individuals who are deaf or hard of hearing, have been largely overlooked by governmental efforts. Contributing factors to the rising prevalence of hearing impairments, encompasses not only the needs of the deaf and hard-of-hearing community but also considers factors such as the aging population and environmental influences, and untreated ear infections. Unfortunately, due to the expensive nature and restricted availability of hearing aids and assistive devices, only a small portion of individuals with hearing impairments can obtain these crucial resources. Consequently, individuals who are deaf or hard of hearing, in India faces considerable obstacles in accessing vital services, education, and social interactions due to communication barriers, as sign language often serves as their main method of communication. [2]

The employment landscape further underscores the obstacles encountered by people with hearing impairments, as indicated by the 2011 census, where 73.9% of the population of individuals aged 15 to 59 in India with hearing impairment was categorized as marginal laborers [1], leaving only 26.1% actively employed. However, advancements in technology offer promising avenues for overcoming these barriers. Innovations in speech-to-text technology, like real-time transcription and automatic closed captioning can greatly improve access to spoken language for the deaf and hard-of-hearing community, facilitating participation in dialogues and meetings. Additionally, machine learning algorithms applied in sign language recognition systems hold promise in narrowing the communication divide between the deaf and hearing communities. These systems have the capability to translate sign language into spoken or written language, facilitating seamless communication.

In this paper, we delve into the a wide range of technologies developed to assist individuals with hearing impairments, emphasizing their capacity to enhance communication and increase accessibility to services, particularly for ASL (American Sign Language) users. We also explore the obstacles related to the creation and integration of these technologies, including considerations of precision, reliability,

Fig. 1: American Word Level Sign Language.[3]

and accessibility. We argue that ongoing investment in research and development is essential to guarantee that technology effectively addresses the requirements of the deaf and hard-of-hearing population, fostering greater inclusivity and accessibility for all. Furthermore, our paper presents the Dual Mode Sign Language Communication Module, that not only acknowledges sign language but also converts text into ASL, thereby facilitating communication for individuals with varying levels of sign language proficiency or hearing abilities. By establishing this comprehensive framework, we aim to facilitate effective communication and inclusivity for individuals who are deaf or hard of hearing on both ends of the communication spectrum.

2. BACKGROUND

Sign language, as a multifaceted and emotionally expressive means of communication, serves as a versatile mode with diverse capabilities However, the widespread lack of proficiency among hearing individuals poses significant barriers to effective communication, particularly for those who experiencing hearing impairment or deafness [2]. This communication gap can restrict their full participation in society.In response to this obstacle, researchers are actively working on automated techniques for translating sign language into text or speech, which could potentially transform interactions between the deaf and hearing communities. The emergence of real-time sign language to text translation devices offers great hope for improving the lives of those with hearing impairments or deafness.

In the American community, American Sign Language (ASL) serves as the primary means of communication for the deaf and hard-of-hearing population. Unlike spoken languages, ASL conveys meaning through intricate hand gestures, facial expressions, and body movements. ASL sets itself apart from spoken languages by communicating through intricate hand gestures, facial expressions, and body movements. Its roots trace back to the early 19th century, growing organically as a form of communication within the American deaf community. Initially introduced by Thomas Hopkins Gallaudet and Laurent Clerc, ASL gained prominence as a language distinct from English.[4] Over time, ASL has developed its own syntax,vocabulary as well as grammar embodying the rich linguistic heritage of the American deaf culture. Widely embraced across the nation, ASL plays an important role in empowering millions of deaf citizens in the United States, facilitating access to education, healthcare, employment, and essential services.[5] Despite its widespread usage, recognizing ASL gestures presents significant challenges because of the nuanced and variable nature of sign language expressions.[6] Therefore, there is an urgent need for the development of precise and efficient methods to recognize ASL gestures, holding the potential to significantly improve the well-being of deaf individuals in America.

Modern methods for recognizing American Sign Language (ASL) utilize advanced computer vision and machine learning technologies to automatically identify gestures in sign language from video recordings. By utilizing computer vision algorithms to track hand,face expression and body movements in videos and extract pertinent features, researchers can discern specific signs. The algorithms including neural networks, are subsequently trained on these features to recognize a broad spectrum of signals in video data.[7] Although these techniques have demonstrated encouraging outcomes, they face hurdles because of the inherent variability and complexity present in sign language gestures. This complexity can impede accurate recognition across different contexts and signers. Furthermore, the lack of universally accepted data sets and evaluation standards adds complexity to comparing and evaluating different ASL recognition approaches. Additionally,Huge amount of annotated data are required for the training of machine learning model in most of the existing

approaches,leading to resource-intensive data collection efforts.[8] Despite these challenges, The latest developments in computer vision and machine learning are driving advancements in ASL recognition, setting the stage for the development of more accurate and effective methods in the future.

3. RELATED WORKS

The integration of sign language into multilingual text and voice output represents an innovative convergence of technology, linguistic accessibility, and inclusivity. For individuals who are hard of hearing or deaf , sign language serves as a vital mode of communication, offering them access to the broader world. However, challenges arise when bridging sign language with spoken and written languages, hindering daily interactions, educational pursuits, and professional endeavors.

In response to these challenges, innovative solutions have emerged, leveraging technology to narrow the communication divide. These applications aim to enhance the accessibility, comprehension, and inclusivity of sign language across diverse linguistic landscapes. By harnessing cutting-edge advancements in natural language processing, computer vision, and machine learning, these applications redefine the integration of sign language into global culture.

Various techniques and methodologies have been developed by researchers to facilitate sign language recognition. For instance, Recurrent Neural Networks (RNNs) are commonly employed for systems reliant on sequential data [10]. Within RNNs, the Long Short-term Memory (LSTM) architecture has gained prominence for mitigating the vanishing gradient problem encountered in traditional RNNs [10]. Notably, a study by [10] utilized the LSTM model to achieve high accuracy in Indian sign language recognition.

Visual-based methods, particularly those employing cameras to capture finger motions, represent a prevalent approach in sign language recognition systems [17]. Extensive efforts have been dedicated to the advancement of vision-based sign recognition systems globally.

In recent years, deep learning has garnered increasing interest for its application in various domains, including sign language recognition [19]. Numerous studies have explored deep learning techniques for classifying images or videos, leveraging the attributes of motion recognition and time series language translation inherent in sign language. [19].

Multilayer perceptron (MLP), a deep neural network architecture comprising input, hidden, and output layers, has been utilized for sign language recognition [16]. Researchers have employed various picture-capturing tools, such as cameras, webcams, data gloves, Kinect, and jump controls, with cameras being the preferred choice due to their natural interaction and simplicity [13].

An overview provided by [18] outlines a methodology consisting of three main modules: feature extraction, processing, and classification. The feature extraction module utilizes MMDetection to detect hand or body bounding boxes, which are then processed using HR-Net, a CNN-based model, to determine key points normalized for gesture identification [18].

Overall, these methodologies underscore the diverse approaches and technologies employed in sign language recognition research, each contributing to the advancement of inclusive communication systems for individuals who are deaf or hard of hearing.

1. **Gesture Recognition with MediaPipe**

   Google's MediaPipe framework offers a versatile platform for constructing machine learning and computer vision pipelines tailored to multimedia applications. Leveraging MediaPipe's pre-built components and tools, developers can create robust pipelinesfor identifying both static and dynamic gestures in Filipino Sign Language [10]. The initial step involves the generation of a dataset, given the absence of publicly available datasets for sign language words. Utilizing a webcam, signs are captured, forming the basis of the dataset. Commonly used words, including single-handed gestures like 'okay,' 'yes,' and 'thumbs up,' as well as two-handed gestures like 'alright' and 'hello,' are included. Following stable camera capture resulting in 2536 images, these images are converted into frames and divided into 75% for training and 25% for testing purposes. Subsequently, video frames are fed into the MediaPipe framework, with Google's MediaPipe Hands offering precise hand and finger tracking capabilities. By employing machine learning algorithms, MediaPipe Hands deduces 21 3D hand landmarks from a single frame. Notably, the framework ensures real-time performance, even on mobile devices, and accommodates multiple hands [11].

2. **Gesture Recognition with LSTM**

   Long Short-Term Memory (LSTM) networks, a variant of recurrent neural networks, are adept at handling sequential data by addressing the challenge of long-term dependencies. Unlike traditional RNNs, which struggle with predicting words stored in long-term memory as the gap length increases, LSTM networks excel at retaining information over extended periods. This

Table 1: Related Works Summary[9]

Ref No	Paper Name	Algorithm & Accuracy	Limitations
10	Long Short-Term Memory based Static and Dynamic Filipino Sign Language Recognition	LSTM Neural Network - 98% Accuracy	Unable to recognize rare or complex gestures.
11	Hand Gesture based Sign Language Recognition using Deep Learning	AlexNet Classifier - 99% Accuracy	Limited only to finger-spelling recognition, which is a subset of sign language.
12	Deep Learning based Sign Language Recognition robust to Sensor Displacement	CNN & LSTM - 95.6% Accuracy	Only works with massive amount of data.
13	A Review of Segmentation and Recognition Techniques for Indian Sign Language using Machine Learning and Computer Vision	CNN	The datasets used are limited in size and availability.
14	Speech to Sign Language Translation for Indian Languages	Wavelet based MFCC & LSTM - 80% accuracy	Voice input is noisy and accuracy is less.
15	Sign Language to Text Conversion using Deep Learning	CNN - 99% accuracy	Model accuracy is less with limited dataset.
16	Sign Language Translator using ML	KNN, Decision Tree Classifier - 97% accuracy	The model does not take into account the context of the signs.
17	Sign Language to speech translation using ML	CNN - 90% Accuracy	System performance may degrade if sensor is placed in a different location.
18	An improved hand gesture recognition system using key-points and hand bounding boxes	CNN - 95% accuracy	For large images this method is computationally expensive.
19	Dataset Transformation System for Sign Language Recognition Based on Image Classification Network	STmap, CNN-RNN - 99% accuracy	Conversion of skeleton data into an image before training the network, might lead to loss of information.

characteristic makes them ideal for processing, predicting, and classifying time-series data. In the context of sign language recognition, LSTM networks are invaluable for analyzing sequential sign language video sequences. Each video frame serves as a time step, enabling the network to capture temporal patterns and dependencies. Key temporal features, such as hand movements and facial expressions, are extracted and utilized for recognition. Furthermore, LSTMs can be employed in conjunction with convolutional neural networks (CNNs) in a hybrid architecture, capitalizing on CNNs' ability to process spatial features in individual frames. By combining spatial and temporal information, the LSTM-CNN hybrid model enhances sign language recognition accuracy. Training the LSTM model involves labeled sign language data, where video frame sequences are associated with specific signs or phrases. Through training, the LSTM learns to discern the dynamics and context crucial for accurate recognition. Once trained, the model can be deployed for real-time sign language recognition, providing predictions based on user-performed signs.

3. **Image Feature Extraction with CNN**

Convolutional Neural Networks (CNNs) specialize in extracting features from images, rendering them instrumental in sign language recognition systems. In this framework, each frame of a gesture video is treated as an image, allow-

ing CNNs to capture and analyze spatial features such as handshapes and facial expressions. CNNs comprise convolutional layers that apply filters to input images, identifying patterns and features relevant to sign language recognition. Pooling layers further refine feature maps, reducing computational complexity while preserving essential features. These filters are trained to detect key aspects of sign language gestures, aiding the CNN in understanding visual characteristics. To augment the robustness of the CNN model, preprocessing techniques such as image resizing and normalization are applied. Additionally, CNNs are often integrated with LSTM networks in a hybrid architecture to consider both spatial and temporal information for recognition. Training the CNN model involves labeled sign language image data, optimizing model parameters and employing techniques to minimize recognition errors. The model learns to recognize important visual features and patterns associated with sign language gestures, thereby enhancing recognition accuracy in real-world applications.

4. METHODOLOGY

4.1. Sign To Text Prediction Model

For the sign-to-text prediction model, we engineered a deep learning architecture leveraging Long Short-Term Memory (LSTM) networks. The model configuration comprises a series of LSTM layers, each integrated with dropout regularization to mitigate overfitting. Rectified Linear Unit (ReLU) activation functions were applied to the LSTM layers to introduce non-linearity. The input shape for the model was specifically set to (30, 1662), symbolizing 30 frames with 1662 keypoints per frame meticulously extracted from sign language videos. This model architecture is structured as follows:

1. The first LSTM layer consists of 64 units, configured to return sequences and activated using ReLU. It processes input sequences with dimensions (30, 1662).

2. A dropout layer, incorporating a dropout rate of 0.3 is introduced after the first LSTM layer to prevent overfitting.

3. The second LSTM layer contains 128 units, also set to return sequences and activated using ReLU.

4. Another dropout layer with a dropout rate of 0.3 follows 2nd LSTM layer for regularization.

5. The 3rd LSTM layer consists of 64 units and is configured to return only the final output sequence, not sequences.

6. Following the third LSTM layer, a dropout layer with a dropout rate of 0.3 is included for regularization purposes.

7. Two fully connected Dense layers with ReLU activation functions, comprising 64 and 32 units respectively, are appended to LSTM layers.

8. The final Dense layer incorporates the softmax activation function to produce probabilities across all possible output classes, which aligns with the quantity of actions represented by the "actions.shape[0]" parameter.

```
model = Sequential([
    LSTM(64, return_sequences = True, activation ='relu', input_shape=(30, 1662)),
    Dropout(0.3),
    LSTM(128, return_sequences = True, activation ='relu'),
    Dropout(0.3),
    LSTM(64, return_sequences = False, activation ='relu'),
    Dropout(0.3),
    Dense(64, activation ='relu'),
    Dense(32, activation ='relu'),
    Dense(actions.shape[0], activation='softmax')
])
```

Fig. 2: LSTM Model Architecture.

4.1.1. Dataset collection

For the data collection phase, our initial step involved gathering essential training and evaluation data for our models. We utilized a blend of MediaPipe's Holistic toolset and OpenCV, along with a webcam arrangement, for recording video sequences depicting individuals performing sign language gestures. MediaPipe Holistic was instrumental in the estimation of pose, hand, and facial landmarks, facilitating the recognition of diverse hand and body movements linked to sign language. Our data collection process occurred in real-time and focused on a predefined repertoire of sign language gestures. To conserve storage resources, we opted not to save the video data directly. Instead, we meticulously stored an array containing the pertinent essential components extracted from the MediaPipe Holistic framework.

1. **Video Input** MediaPipe Holistic is compatible with both live video feeds from a camera and pre-recorded video files.

2. **Pose Estimation** The initial phase of utilizing MediaPipe Holistic involves identifying the pose of the individual within the video frame. This process entails locating critical body components like the head, shoulders, arms, hips, and legs. BlazePose, a machine learning model, is employed by MediaPipe Holistic for this task.

3. **Estimation of Landmarks on Hands and Faces** Following pose estimation, the subsequent

step is to detect the landmarks on the face and hands. MediaPipe Holistic utilizes separate machine learning models dedicated to estimating hand and facial landmarks.The facial landmark model detects 468 crucial points on the face, whereas the hand landmark model identifies 21 key points on each hand.

4. **Incorporation of Keypoints** Once the landmarks on the hands, face, and body are identified, MediaPipe Holistic amalgamates these key points to generate a holistic representation of the individual within the video frame. This amalgamation of key points into a unified coordinate system facilitates precise tracking of the individual's movements.

5. **Output** MediaPipe Holistic produces a collection of key points or landmarks that depict the pose, hand gestures, and facial expressions of the individual captured within the video frame.

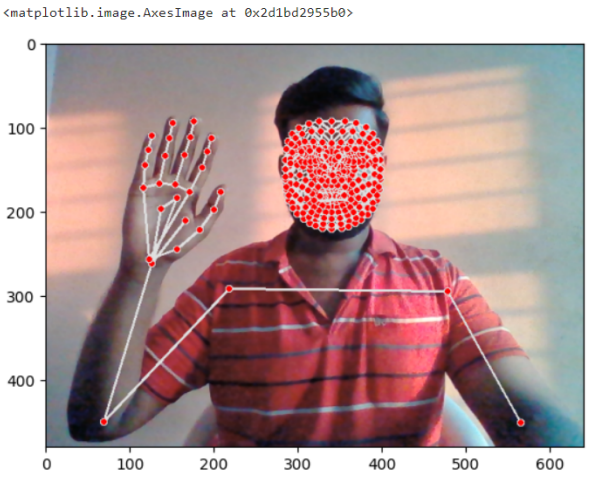

Fig. 3: Sample Data Collection using MediaPipe Holistic

MediaPipe Holistic is employed to analyze video frames and identify the pose, hand gestures, and facial landmarks of an individual. By amalgamating these key points, it constructs a holistic portrayal of the person in the video frame, facilitating accurate monitoring of their actions.

In each frame, a comprehensive set of 1662 key points, derived from pose, hand, and facial landmarks, is compiled into a unified coordinate space. In instances where certain points are not detected by the model, we input zero values to maintain the array's structure.

For this, we manually recorded sign language videos, capturing 30 videos for each of the 10 gesture classes: 'GoodBye', 'Catch', 'Crown', 'Dance', 'Hello', 'None', 'Phone', 'Please', 'Thank-You', and 'Yes'. Each video

```
[108]: np.array(sequences).shape

[108]: (300, 30, 1662)
```

Fig. 4: LSTM Model Shape.

was segmented into 30 frames, and a total of 1662 keypoints were extracted from each frame using pose estimation techniques. The dataset dimensions were (300, 30, 1662), denoting 300 samples, each comprising 30 frames with 1662 keypoints per frame. Subsequently, we constructed a label map for the sequences, which will serve as the basis for training the dataset.

```
[106]: label_map = {label:num for num, label in enumerate(actions)}
       label_map

[106]: {'GoodBye': 0,
        'Catch': 1,
        'Crown': 2,
        'Dance': 3,
        'Hello': 4,
        'None': 5,
        'Phone': 6,
        'Please': 7,
        'Thank-You': 8,
        'Yes': 9}
```

Fig. 5: LSTM Model Label Map.

4.1.2. Model

The training phase is crucial in developing an efficient system for recognizing sign language, utilizing advanced machine learning techniques. This section discusses the training strategies implemented for the models showcased in this project. It involves training on two distinct datasets and compares the effectiveness of time series models like LSTM with a CNN model, known for its proficiency in image processing tasks.

Before initiating model training, thorough data preprocessing steps were conducted to ensure the data's compatibility with the training process. These preprocessing procedures involved extracting essential features from every frame of the sign language videos, followed by standardizing their array size through resizing operations. Additionally, the datasets were carefully divided into 90:10 training and testing splits, a common practice aimed at supporting robust model training and ensuring accurate assessment of the models' performance in predicting sign language expressions.

LSTM Model

The LSTM model architecture starts with a 64-unit LSTM layer designed to process input sequences with 30 timesteps and 1662 features. ReLU acts as the activation function for this LSTM layer, enabling nonlinear transformations within the network. The return sequences parameter is set to True, indicating that the layer will produce sequences rather than just the final output. Batch normalization is then applied after each LSTM layer to address the vanishing gradient problem and improve training convergence.

After each batch normalization layer, a dropout layer with a dropout rate of 0.3 is added to prevent overfitting. Additionally, L2 regularization is used on the kernel weights of the second LSTM layer, which has 128 units. The third LSTM layer has 64 units and is set to return sequences as False, providing only the final output of the sequence.

Two dense layers follow, both using ReLU activation functions. The first dense layer has 64 units, and the second has 32 units. Each dense layer includes a kernel regularizer parameter for L2 regularization.

The last dense layer has 26 units, matching the number of classes in the dataset for multi-class classification. The softmax activation function is used in this layer. The model is optimized with the Adam optimizer and a learning rate of 0.001, and categorical cross-entropy is the chosen loss function for training progress computation.

```
[125]: model = Sequential([
           LSTM(64, return_sequences = True, activation ='relu', input_shape=(30, 1662)),
           Dropout(0.3),
           LSTM(128, return_sequences = True, activation ='relu'),
           Dropout(0.3),
           LSTM(64, return_sequences = False, activation ='relu'),
           Dropout(0.3),
           Dense(64, activation ='relu'),
           Dense(32, activation ='relu'),
           Dense(actions.shape[0], activation='softmax')
       ])

[126]: model.compile(optimizer = 'Adam', loss = 'categorical_crossentropy', metrics=['categorical_accuracy'])

[127]: class myCallback(tf.keras.callbacks.Callback) :
           def on_epoch_end(self, epoch, logs={}) :
               accuracy = logs.get('categorical_accuracy')
               if(accuracy is not None and accuracy >= 0.95) :
                   print("\n Reached 95% accuracy... So cancelling training!!!")
                   self.model.stop_training = True
       mycallback = myCallback()

[128]: model.fit(X_train, Y_train, epochs=250, callbacks=[mycallback, tb_callback])
```

Fig. 6: LSTM Model Used.

Accuracy is the primary evaluation metric during training. Callbacks are utilized for specific actions at defined training stages. A learning rate scheduler dynamically adjusts the optimizer's learning rate, and early stopping halts training if the model's performance on the validation set doesn't improve after a set number of epochs. TensorBoard notifications are employed for training documentation and result visualization.

Training is initiated using the fit method of the model object, with training data and labels provided. Validation data and labels are also used to monitor model performance. The model undergoes 250 epochs for training.

4.1.3. Model Evaluation

In assessing the effectiveness of our model on the assessment dataset, we focused on measuring precision as a key metric. Additionally, we generated confusion matrices to gain deeper insights into the model's performance across different classes. By contrasting the efficacy of different versions of our model, Our aim was to identify the most effective architecture for each dataset.

Our model architecture comprises a combined count of 596,906 trainable parameters, indicating its capacity to learn from the data effectively. Notably, all parameters are trainable, indicating that the model can adapt to the nuances of the gesture recognition task. In our evaluation, we found that our model demonstrated high precision across various sign language recognition tasks. The precision metric served as a reliable measure of the model's ability to accurately classify sign language gestures.

Moreover, the confusion matrices offered significant insights into the performance of the model, illuminating strengths and pinpointing potential areas for enhancement. By analyzing the confusion matrices, we gained a comprehensive understanding of how our model performed across different sign language classes.

Overall, our study highlights the significance of choosing suitable model architectures tailored to particular tasks in gesture recognition. Through the evaluation of various model variants, we pinpointed the most efficient architecture for each dataset, thus furthering the progress of gesture recognition technology.

4.1.4. Utilization of the model and algorithms

In our project focusing on gesture recognition, the foundation lays in the making of a comprehensive dataset capturing both static and gesture-based signs. We meticulously curated a dataset consisting of 10 unique words for gesture-based signs and 26 static signs corresponding to different alphabet classes. Utilizing OpenCV with a webcam and MediaPipe Holistic, we tracked and visualized key points of hand gestures, enhancing the recognition process.

Once the body gestures were captured, we organized the key points for each class and video into an array, constituting the dataset utilized for training our ges-

```
[129]: model.summary()

Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_18 (LSTM)              (None, 30, 64)            442112

 dropout_18 (Dropout)        (None, 30, 64)            0

 lstm_19 (LSTM)              (None, 30, 128)           98816

 dropout_19 (Dropout)        (None, 30, 128)           0

 lstm_20 (LSTM)              (None, 64)                49408

 dropout_20 (Dropout)        (None, 64)                0

 dense_18 (Dense)            (None, 64)                4160

 dense_19 (Dense)            (None, 32)                2080

 dense_20 (Dense)            (None, 10)                330

=================================================================
Total params: 596906 (2.28 MB)
Trainable params: 596906 (2.28 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Fig. 7: LSTM Model Summary.

ture recognition model. With the dataset prepared, we proceeded to train our machine learning model, leveraging Long Short Term Memory (LSTMs) Neural Network to effectively learn and classify sign language gestures.

Following the training phase, we seamlessly integrated our model into a web application built using the MERN stack, creating a video call platform. This platform featured two distinct sections for ASL communication: one for converting gestures to text and another for converting text to ASL images. To facilitate speech-to-text conversion, we incorporated a speech recognition API capable of accurately deciphering spoken words into individual components.

During communication sessions on our platform, when a deaf user showcased a sign, it was promptly converted to text and displayed on the screen of the hearing user. Conversely, when a hearing user spoke, their words were converted into 2D images of ASL gestures and displayed on the screen of the deaf user.

By synergizing OpenCV, MediaPipe Holistic, CNNs, and a speech recognition API, we successfully developed a system for gesture recognition and conversion. This comprehensive approach has significant promise in improving communication and narrowing the divide between the deaf and hard-of-hearing community and the general public, promoting inclusiveness and comprehension.

4.2. Text To 2D Sign Conversion Algorithm

The text-to-2D sign algorithm enables real-time translation of spoken language, which is then converted to text, and then to American Sign Language (ASL). It begins by capturing user voice, which undergoes speech recognition to form an English sentence. This sentence is parsed into words, generating a sequence for ASL translation. Through regular expression extraction and sequence generation, each word is translated into ASL gestures using the Sign Producer module. The final output consists of ASL images, facilitating communication for ASL users.

4.2.1. Algorithm Architecture

The text-to-2D sign architecture encompasses seven distinct stages, each playing a crucial role in the translation process, as shown in the figure [9] [20]:

1. **User Voice:** The initial input to the system is the user's voice, which contains the spoken language to be translated into American Sign Language (ASL).
2. **Speech Recognition Module:** This module utilizes advanced algorithms to transcribe the user's speech into a string sentence in English. The transcribed sentence serves as the linguistic basis for further processing.
3. **String Sentence:** The transcribed English sentence represents the textual form of the spoken language and serves as the primary input for subsequent stages.
4. **Regular Expression Extraction:** This stage involves the use of regular expression techniques for sentence parsing and word separation. The transcribed sentence is parsed, and individual words are extracted to form a sequence for further processing.
5. **Sequence Generator:** Using the extracted sequence of words, this stage generates a structured input for the sign production process. Each word in the sequence serves as a unit of meaning for the subsequent ASL translation.
6. **Sign Producer:** This stage processes the ASL sequence by generating word tokens. For each token, it checks for its presence. If found, it adds it to the playing sequence. If not found, it handles exceptions by breaking the word token into letters. Each letter token is then processed similarly. If a letter token is not found, an exception is raised. This process continues until all word tokens are processed, ensuring accurate ASL representation, the architecture of this stage can be seen in the figure [10].
7. **ASL Images:** The final output of the system comprises a sequence of ASL images, visually representing the translated sign language. These images serve as the direct output of the text-to-2D sign algorithm and facilitate communication for individuals proficient in ASL.

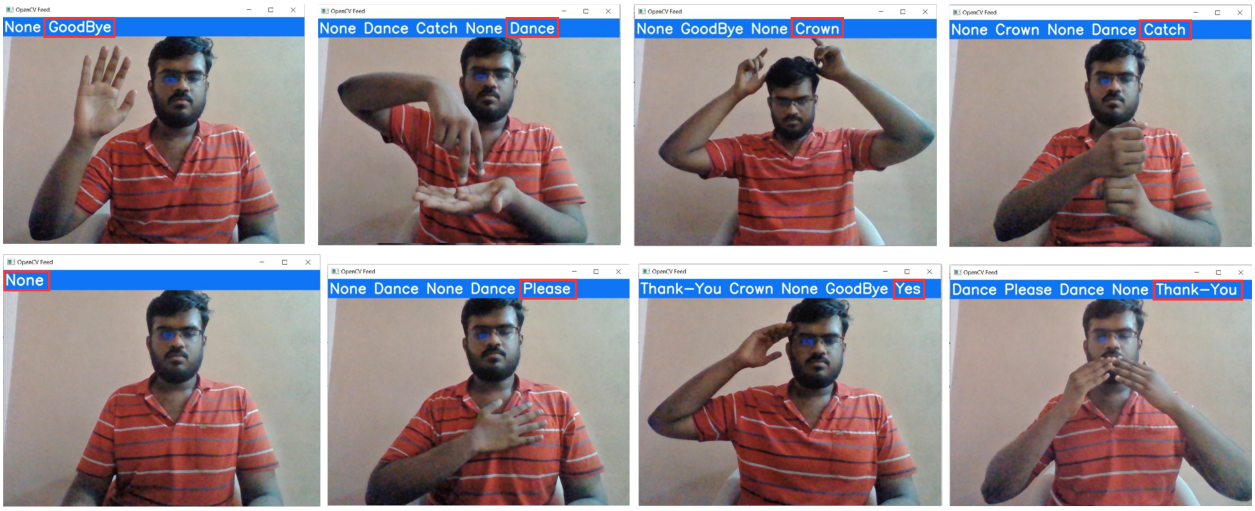

Fig. 8: Live Sign Language Detection Feed

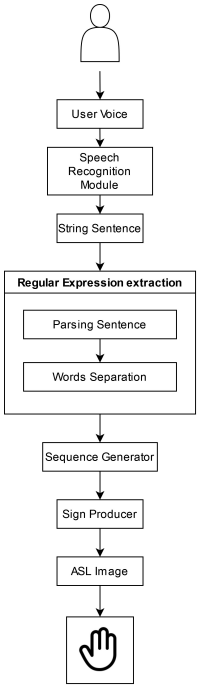

Fig. 9: Architecture of Text To 2D Sign Conversion Algorithm

4.2.2. Sign Producer Architecture

In the Sign Producer stage, the module receives an ASL sequence, a list of words, as input. It begins by generating word tokens from the sequence, iterating through each token to determine its presence. If a word token is found, it adds it to the playing sequence. However, if a word token is not found, the module handles the exception by breaking the token into letters. Subsequently, it iterates through each letter token, checking for its presence. Found letters are added to the playing sequence, while any letter token not found raises an exception. This process continues until all word tokens are processed. If there are remaining letter tokens, the process repeats until each token is accounted for. Once all word tokens are processed, the module concludes the process, ensuring accurate ASL representation for the given sequence, this can be seen in the figure [10] [20].

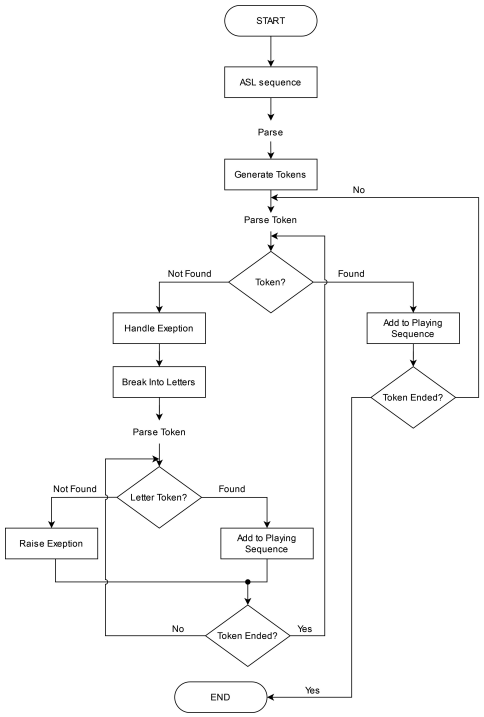

Fig. 10: Sign Producer Architecture

4.2.3. **Database**

The Sign Producer stage relies on Firebase, as shown in figure [11], as its database for managing ASL data. Firebase offers cloud storage, enabling access from anywhere without local storage requirements, thereby preventing unnecessary increases in the web application's size. Moreover, Firebase's scalability allows seamless expansion of the database to accommodate a growing library of ASL words. This scalability ensures the system's capability to identify and translate an extensive range of words into ASL gestures. By leveraging Firebase, the Sign Producer module efficiently handles the storage and retrieval of ASL data, contributing to the system's overall efficiency and flexibility. The integration with Firebase enhances the user experience by providing reliable and adaptable storage solutions for ASL communication. This choice reflects a strategic decision to optimize storage accessibility and scalability, supporting the system's capacity to effectively interpret and convey spoken language into ASL representations.

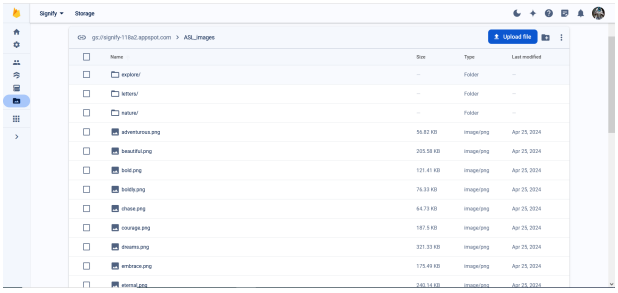

Fig. 11: Database, source: FireBase

5. RESULTS

5.0.1. **Sign To Text Prediction Model**

Our model for sign language recognition demonstrated outstanding performance, achieving a categorical accuracy of 97.89%. This remarkable accuracy highlights the effectiveness of our approach in precisely identifying sign language gestures.

```
[33]: print("Test Accuracy", accuracy_score(ytrue, yhat))
      print("Precision:", precision)
      print("Recall:", recall)
      print("F1 Score:", f1)

Test Accuracy 0.9333333333333333
Precision: 1.0
Recall: 0.9333333333333333
F1 Score: 0.9555555555555555
```

Fig. 12: LSTM Model Performance.

Furthermore, our model showcased exceptional precision, achieving a score of 1.0. This indicates consistent accuracy in our model's predictions, with minimal false positives. The recall metric, assessing the model's capability to identify relevant instances accurately, also performed impressively at 0.933. This indicates that our model effectively captured a significant portion of true positive instances. Moreover, the F1 Score, a comprehensive metric combining precision and recall, resulted in a high score of 0.956. This balanced measure further validates the robustness of our model's performance.

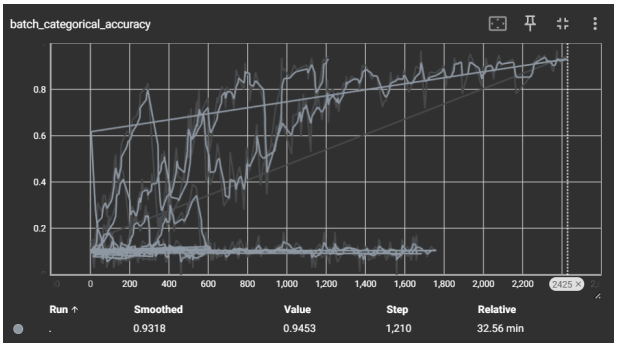

Fig. 13: Model's Batch Categorical Accuracy.

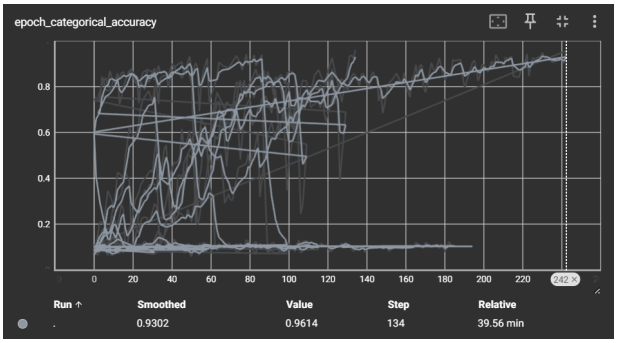

Fig. 14: Model's Epoch Categorical Accuracy.

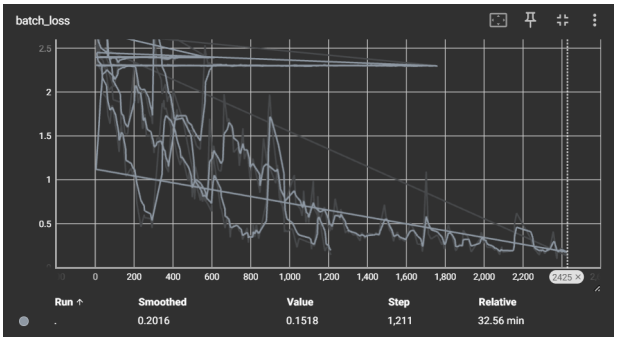

Fig. 15: Model's Batch Loss.

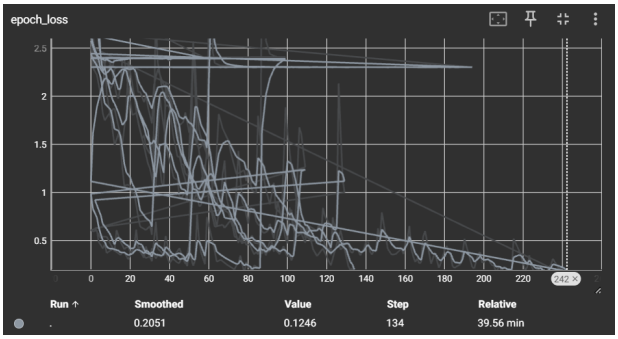

Fig. 16: Model's Epoch Loss.

Analysis of the model's training process revealed promising results. Both batch and epoch categorical accuracies consistently improved throughout the training process, indicating that the model was learning and improving over time. Similarly, both batch and epoch losses steadily decreased, signifying that the model was effectively minimizing errors and optimizing performance.

Furthermore, examination of the confusion matrix, represented as a heat map, provided valuable insights into the model's performance across the ten output categories. Despite the complexity of the task, our model demonstrated remarkable accuracy, correctly predicting eight out of ten gestures with high confidence. Remarkably, our model achieved remarkable accuracy within 150 epochs, with 97.89% categorical accuracy attained. This underscores the efficiency and effectiveness of our training approach, delivering impressive results within a reasonable timeframe.

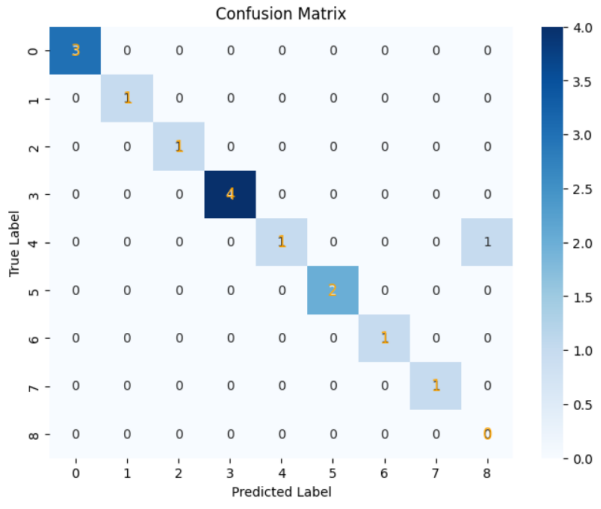

Fig. 17: Confusion Matrix.

In conclusion, our sign language recognition model showcases exceptional performance, accurately identifying sign language gestures with high precision, recall, and F1 Score. These results underscore the potential of our approach to facilitate seamless communication and foster inclusivity for the deaf and hard-of-hearing community.

5.0.2. Text-To-2D Sign Conversion Algorithm

The text-to-2D sign algorithm functions as a sophisticated system aimed at facilitating seamless communication between spoken language and American Sign Language (ASL). It involves several stages, starting with the user's input through spoken language, which undergoes transcription and parsing to generate a sequence of English words. This sequence serves as the basis for ASL translation, where each word is converted into its corresponding ASL sign or gesture. The resulting sequence of ASL images visually represents the translated sign language, enabling effective communication for individuals proficient in ASL.

To utilize the text-to-2D sign algorithm, the user begins by speaking into a microphone, providing the input for the algorithm. Subsequently, the algorithm returns the sequences of ASL images one after the other, with each image displayed for 2 seconds before transitioning to the next. In cases where a word is not found in the database, the algorithm displays the letters of the word sequentially, each for 2 seconds, before moving on to the next word. This interactive process ensures a smooth and comprehensible conversion of spoken language into ASL representations as shown in the figure [18]

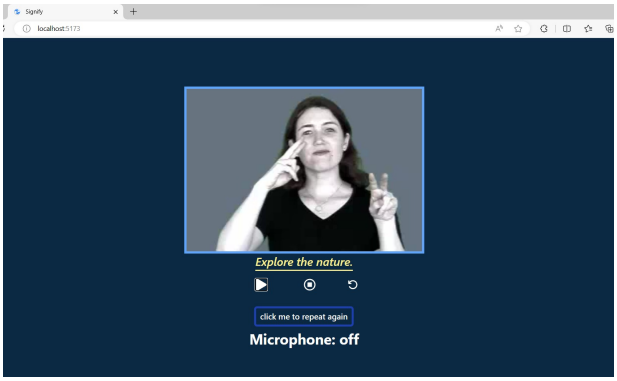

Fig. 18: text-to-2D sign Output

In conclusion, the success of the algorithm in accurately converting text into ASL sign language signifies a significant advancement in inclusive communication. By bridging the gap between spoken language and ASL, this technology enhances accessibility for deaf individuals, enabling them to better understand and engage with hearing users. The implementation of such a model demonstrates a commitment to inclusivity and underscores the potential of technology to

foster greater understanding and connection among diverse communities.

6. FUTURE WORKS

Looking ahead, the quest for enhancing accessibility for the deaf and hard-of-hearing community through sign language recognition technology remains an ongoing endeavor, with numerous avenues for future exploration and innovation.

One promising avenue for future research involves the exploration of augmented reality (AR) and virtual reality (VR) technologies to augment the acquisition and utilization of sign language, particularly in complex environmental settings. By leveraging AR and VR platforms, users could immerse themselves in simulated environments where they can practice and refine their sign language skills in a more interactive and engaging manner. Such immersive experiences could facilitate more effective learning and communication for individuals within the deaf and hard-of-hearing community.

Furthermore, there is a critical need to enhance the accuracy and reliability of sign language recognition systems, especially for complex gestures inherent in sign languages like Indian Sign Language. Future research endeavors could focus on developing novel machine learning algorithms specifically tailored to the nuances of Indian Sign Language, thereby improving the system's ability to accurately interpret and translate a diverse range of signs. Additionally, advancements in data capture and annotation methodologies are essential to ensure that recognition systems are trained on comprehensive and representative datasets encompassing various sign variations and environmental conditions.

Additionally, future investigations should explore the social and cultural factors that impact the adoption and use of sign language recognition systems within the Indian context. Understanding the attitudes, perceptions, and socio-cultural barriers encountered by the deaf and hard-of-hearing communities is crucial for developing technology solutions that are accessible and efficient. By addressing these societal and cultural factors, future research can lead to the development of more inclusive and user-centric sign language recognition technologies customized to meet the specific needs and preferences of Indian Sign Language users.

In conclusion, continuous research and development in technology-based solutions for Indian Sign Language offer significant potential to improve the quality of life and accessibility for India's deaf population. Embracing emerging technologies, refining machine learning algorithms, and gaining a deeper understanding of socio-cultural dynamics can pave the way toward a more inclusive and empowered future for the deaf and hard-of-hearing community in India.

7. CONCLUSION

In conclusion, our efforts to bridge communication barriers for the deaf and hard-of-hearing community have resulted in a comprehensive dual-mode sign language communication framework. By integrating advanced technologies such as OpenCV, MediaPipe, LSTM networks, and speech recognition APIs, we've developed a versatile system facilitating seamless interaction between sign language and text-based communication. Our LSTM-based sign-to-text prediction model achieves remarkable accuracy, with a categorical accuracy of 97.89% and perfect precision for gesture recognition, empowering individuals within the deaf community to express their thoughts and ideas effectively.

Complementing this achievement is our text-to-2D sign algorithm, converting spoken language into visually expressive sign language gestures. This platform enhances accessibility and fosters inclusivity and understanding in society. Reflecting on our journey, we acknowledge technology's profound impact in fostering empathy, understanding, and connection among diverse linguistic backgrounds. Moving forward, we're committed to advancing sign language recognition technology, exploring new avenues like augmented reality and virtual reality for enhanced accessibility and effectiveness.

Embracing innovation and collaboration, we envision a future where communication transcends barriers, ensuring every voice is heard and understood. Our goal is to create a world that is inclusive and interconnected, where sign language serves as a bridge for empathy and unity among all.

8. Citations and references

1. Wikipedia contributors. (2023, March 23). Deafness in India. In Wikipedia, The Free Encyclopedia. Retrieved 10:49, April 15, 2023, from https://en.wikipedia.org/w/index.php?title=Deafness_in_India&oldid=1146276949

2. Velmula, K.R., Linginani, I., Reddy, K.B., Meghana, P., Aruna, A. (2021). Indian Sign Language Recognition Using Convolutional Neural Networks. In: Kiran Mai, C., Kiranmayee, B.V., Favorskaya, M.N., Chandra Satapathy, S., Raju, K.S. (eds) Proceedings of International Conference on Advances in Computer Engineering and Communication Systems. Learning and Analytics in Intelligent Systems, vol

20. Springer, Singapore. https://doi.org/10.1007/978-981-15-9293-5_35

3. Tolentino, Lean Karlo & Serfa Juan, Ronnie & Thio-ac, August & Pamahoy, Maria & Forteza, Joni & Garcia, Xavier. (2019). Static Sign Language Recognition Using Deep Learning.*International Journal of Machine Learning and Computing. 9. 821-827. 10.18178/ijmlc.2019.9.6.879.*

4. Gallaudet, T. H., & Clerc, L. (1817). American Sign Language History

5. National Institute on Deafness and Other Communication Disorders (NIDCD). (n.d.). American Sign Language (ASL).

6. Mitchell, R. E., Young, T. A., Bachleda, B., & Karchmer, M. A. (2006). How Many People Use ASL in the United States? Why Estimates Need Updating. Sign Language Studies, 6(3), 306–335.

7. Starner, T., & Pentland, A. (1995). Real-time American Sign Language recognition from video using hidden Markov models. Proceedings of the International Workshop on Automatic Face- and Gesture-Recognition.

8. Starner, T., & Pentland, A. (1997). Visual recognition of American Sign Language using hidden Markov models. *International Journal of Computer Vision, 14(1), 73–98.*

9. Karanjkar, Vaishnavi & Bagul, Rutuja & Singh, Raj & Shirke, Rushali. (2023). A Survey of Sign Language Recognition. *INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT. 07. 1-11. 10.55041/IJSREM26316.*

10. Carmela Louise L. Evangelista, Criss Jericho R. Geli, Marc Marion V. Castillo: Long Short-Term Memory-based Static and Dynamic Filipino Sign Language Recognition (2023)

11. Roli Kushwaha, Gurjit Kaur, Manjeet Kumar: Hand Gesture Based Sign Language Recognition Using Deep Learning (2023)

12. Rinki Gupta, Roohika Manodeep Dadwal: Deep Learning based Sign Language Recognition robust to Sensor Displacement (2023)

13. Subhangi Kumari, Ernest Tarlue, Aissatou Diallo, Megha Chhabra, Gouri Shankar Mishra, Mayank Kumar Goyal: A Review of Segmentation and Recognition Techniques for Indian Sign Language using Machine Learning and Computer Vision (2023)

14. Jashwanth Peguda, V Sai Sriharsha Santosh, Y Vijayalata, Ashlin Deepa R N, Vaddi Mounish: Speech to Sign Language Translation for Indian Languages (2022)

15. Dr. Aruna Bhat, Vinay Yadav, Vishesh Dargan, Yash: Sign Language to Text Conversion using Deep Learning (2022)

16. Jaya Nirmala: Sign language translator using machine learning (2022)

17. Mrs.Aerpula Swetha, Vamja Pooja, Vundi Vedavyas, Challa Datha Venkata Naga Sai Kiran, Sadu Sravan: SIGN LANGUAGE TO SPEECH TRANSLATION USING MACHINE LEARNING (2022)

18. Tuan Linh Dang[a,*], Sy Dat Tran[a], Thuy Hang Nguyen[a], Suntae Kim[b], Nicolas Monet[b]: "An improved hand gesture recognition system using keypoints and hand bounding boxes" (2022)

19. Sang-Geun Choi , Yeonji Park and Chae-Bong Sohn : Dataset Transformation System for Sign Language Recognition Based on Image Classification Network (2022)

20. Ahmad, Babar & Kashif, Muhammad & Safdar, Tauqeer & Hassan, Mehdi& Hasan, Mohd Hilmi & Aziz, NorShakirah. (2021). A Real-Time Automatic Translation of Text to Sign Language. Computers, Materials and Continua. 70. 2471-2488. 10.32604/cmc.2022.019420.