Received: August 7, 2023. Revised: September 30, 2023.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Recognition of Indian Continuous Sign Language Using Spatio-Temporal Hybrid Cue Network

Prema Muthusamy[1]* **Gomathi Pudupalayam Murugan[1]**

[1]*Department of Computer Science, P.K.R Arts College for Women, Erode- 638476, Tamilnadu, India*
* Corresponding author's Email: mprema.hanuphd@gmail.com

Abstract: People who are deaf or hard of hearing typically communicate primarily through sign language (SL). A complication arises when a deaf\mute person tries to use SL with someone who aren't familiar with it. The use of modern technological advancements will be useful in this situation. In recent years, prediction of SLs has been greatly improved by deep learning (DL) techniques. Amongst, a spatial-temporal multi-cue (STMC) network combines a spatial multi-cue (SMC) module and temporal multi-cue (TMC) to learn spatial and temporal encoding of multi-cue features like full-frame, hands, face, and stance. While it outperforms single-cue algorithms on large SL datasets, it requires more time to pre-process the dataset. Also, annotating key points is required for the complete training of the STMC network. This article presents a spatio-temporal hybrid cue network (STHCN) using dynamic dense spatio-temporal graph convolutional neural network (DDSTGCNN) and VGG11+1D-CNN+BLSTM feature extractor network to address the fore-mentioned issues of ISL recognition and translation tasks. Initially, the skeleton and full frame data is extracted from the input video. The DDSTGCNN is a combination of dense-GCNN (DGCNN) and dynamic spatial-temporal convolution network module (DSTCNM) which learns spatial features temporal features of skeleton data. VGG11+1D-CNN+BLSTM is used to extract the features from full frame data. Then, the extracted features are fed into a bidirectional long-short term memory (BLSTM) encoder, connectionist temporal classification (CTC), and self-attention based LSTM (SA-LSTM) decoders for sequence learning and inference. Finally, The CTC and SA-LSTM predict ISL from input videos and sentences for ISL detection and translation. Finally, an experiment results reveal that the STHCN model achieves 93.65% accuracy in detecting the ISL.

Keywords: Indian sign languages, Convolutional neural network, Long-short term memory, Spatio-temporal convolution module, Graph convolutional neural network.

1. Introduction

Deaf and dum individuals rely heavily on sign language (SL) for communication, as they have little to no auditory perception. SLs consist of hand shapes, actions, orientations, and facial expressions [1]. Nearly 466 million people have hearing loss, including 34 million children. People worldwide use various SLs for perspective communication [2].

Sign language (SL) is a visual language that does not rely on words or sentences. Its signs can be manual or non-manual. Non-manual signals include things like facial gestures, mouth and head actions, and so on, whereas manual signs are things like hand and finger motions, hand direction and movement, and so forth [3]. Many countries use different SLs for communication, leading to a lack of standardization; for example, India utilizes ISL [4] and the United States uses American sign language (ASL) [5]. SL is organized differently in terms of spatial and temporal aspects with phrases often consisting of time, place, person, and predicate.

Sign language recognition (SLR) aims to automate translating signs into spoken or written language, facilitating communication between hearing-impaired individuals and the general audience. This technology can aid HCI-based apps in advance [6]. Researchers have developed single-word SLR systems, but they struggle with constant motion sequences. Developing an adaptive framework to capture sign gestures and related words is the main challenge in creating an autonomous SLR

system [7].

SL is classified into two types: static and dynamic. Dynamic may be further classified into isolated signs and continuous signs, whereas static or isolated signs are unchanging hand and face motions [8]. Though beneficial progress have been achieved in the field of isolated SLR, in which algorithms only needed to detect a particular alphabetic sign, word or sentences which involves the interpretation of extended speech segments. But, this process cannot be confined in recognizing particular gestures since contextual interactions among signals have a significant impact on phrase translation. Videos showing sequences of signs, rather than individual signs, may be found in continuous sign language recognition (CSLR) databases, making them better suited to the development of practical applications [9].

DL models have emerged as an innovative tool for researchers to solve the sign language recognition (SLR) problem. These models use data from various domains to recognize sign language concepts, but their effectiveness depends on the selected dataset [10]. Examples include CNN, recurrent neural network (RNN), long-short time memory (LSTM), and deep belief network (DBN). DL-based applications enable SL translation to text, improving communication between signers and non-signers. However, in SLR tasks like continuous speech interpretation or real-time translation, higher layers may be required. DL models are considered a safe option for CSLR applications, as they efficiently understand semantic aspects of sign communication and improve predictability of SLs for deaf or hearing-impaired individuals [11].

Deep learning models for CSLR systems often focus on highly differentiating features, neglecting non-trivial information. To address this issue, a STMC network [12] was developed combining a SMC module and TMC module in a video-based sequence learning model. The SMC component incorporates a decoupled posture estimation branch for spatial representation of stimuli, while the TMC module mimics temporal corrections from within-cue and between-cue viewpoints. A 2D CNN is used in the spatial encoding module to provide multi-cue features of full-frame, hands, face, and attitude. The STMC model outperforms single-cue techniques on large SL datasets, but requires more time and supervised key-point annotations for full training tasks.

Hence, in this paper, STHCN is proposed using DDSTGCNN and VGG11+1D-CNN+BLSTM feature extractor network to resolve the fore-mentioned issues for ISL recognition and translation tasks. Initially, the skeleton data and full frame data is obtained from the input videos. To extract the skeleton data, skeleton extraction algorithm is used whereas, full frame data is obtained directly from the input videos. The extracted skeleton data is given as input to the DDSTGCNN. DDSTGCNN is combination of DGCNN and DSTCNM. In the DGCNN module, DGCF and STAN are performed to learn the spatial features which pays attention to the key frames information and joints from the intermediate features. In DSTCNM structure, in order to perform recognition tasks effectively, skeletal sequence temporal information is captured using 1D convolution layers. VGG11+1D-CNN+BLSTM is used to extract feature from full frame data. For sequential training and assumption, the features retrieved by DDSTGCNN and VGG11+1D-CNN+BLSTM are fed into the BLSTM encoder along with CTC and SA-LSTM decoders. Finally, the CTC predict the ISL from input videos and sentences from the collected video frame through SA-LSTM for the prediction and translation of ISL.

The residual portions of this manuscript are arranged as: Section 2 reviews the previous research associated with the SL recognition and translation. Section 3 explains the complete framework of STHCN for ISL recognition and translation. Section 4 displays its efficacy. Section 5 summarizes the whole study and suggests further development.

2. Literature survey

Mittal et al. [13] developed a modified LSTM model for detecting and monitoring continuous ISL words using a motion sensor. The data was pre-processed and normalized, then CNN model applied to signed sequences. A modified LSTM classifier identified consecutive gesture sequences for ISL. But, the accuracy was lower when training on larger datasets.

Papastratis et al. [14] developed a cross-approach alignment of video and text embeddings using textual data to describe intra-gloss relationships and produce detailed video-based implicit depictions for CSLR. A jointly generated decoder categorizes the aligned video latent depictions, but overfitting was not addressed properly.

Zhou et al. [15] developed a bert-based DL framework called SIGNBERT for CSLR which combines BERT and ResNet to predict SLs and retrieve spatial features. The multimodal implementation reduces BERT model identification distance and hand image probability ranges, but, this model necessitates more training time.

Natarajan et al. [16] created a hybrid deep neural architecture (H-DNA) framework for real-time ISL

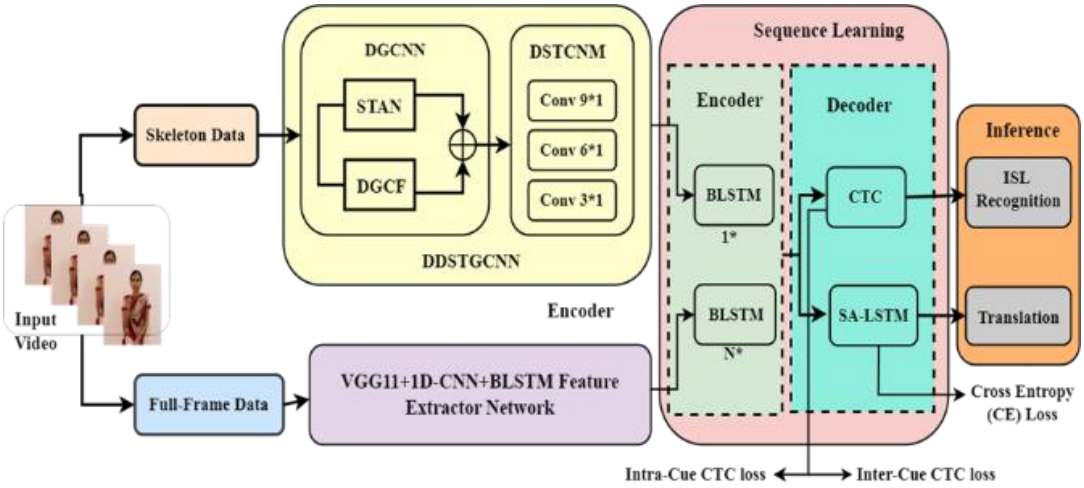

Figure. 1 Schematic representation of proposed STHCN model

detection, translation, and video creation. The framework uses MediaPipe library, CNN, and Bidirectional LSTM models to extract posture information and generate text sequences. But, this model results with lower accuracy and high loss of information.

Katoch et al. [17] presented a model using SVM and CNN to recognize ISL alphabets and digits in live video streams. The model uses the bag of visual words model, segmentation using background removal and skin color and histograms to map signs with matching labels. However, this model results with lower F1-score and slower convergence rate.

Kothadiya et al. [18] developed a DL model to detect words based on gestures in ISL video frames. The InceptionResNetV2 model extracted gesture features and fed them into an RNN for ISL prediction. However, this model was trained with limited dataset and results in high temporal complexity.

Subramanian et al. [19] developed a Mediapipe-optimized gated recurrent unit (MOPGRU) model for ISL recognition involving data pre-processing, feature extraction, keypoint recording, and retraining movements using translated sign motions. However, the non-stable and angular input data frames needed to be focused to enhance accuracy.

Sreemathy et al. [20] developed a CSLR recognition system using a DL model and two hand gesture recognition systems, SVM with media-pipe and YOLOv4. The system compares the two model's accuracy rate determining better real-time training and detection confidence. However, this model provides lower accuracy results on smaller datasets.

3. Proposed methodology

In this section, the complete structure of STHCN is briefly illustrated. The Fig. 1 depicts the schematic representation of the proposed model. The notations used in this study are presented in Table 1.

3.1 Pre-processing

After collecting the database, it is necessary to pre-process input images are collected from web or video cameras. Each image frame is pre-processed to eliminate noise, suppresses unwanted distortions using a variety of filters including erosion, dilation, and Gaussian smoothing, among others. Other important functions are listed below.

- **Detecting blur frames:** The Laplacian approach which examines one frame at a time from a video and generates the focus measure of the frame to eliminate the noise which is used to identify the blurred frames.
- **Cropping images:** The bound reduces each frame of a video to remove irrelevant background, enabling OpenPose [21] to determine keypoint positions. The original video is cropped using the same bound to identify body postures and motions.
- **Rotate and flip frames for better view:** To improve video viewing, turn frames left and right to alter frame alignment. The palm and fingers represent the position of the hand and the signer's body, while the palm can be angled towards the signer, up, down, right, left, or edge towards the signer.

3.2 Skeleton data processing by dynamic dense spatio-temporal graph convolutional neural network (DDSTGCNN)

The skeleton data is extracted from input video using the skeleton extraction technique [22], which involves capturing frames with various human joint coordinates. This data is used to evaluate a specific human's skeleton, as it contains positional data on

Table 1. Lists of notations

Notations	Description
g	Graph
$(v\ ,e)$	Collection of Body Joints and Bones
ℓ	Graph Convolution
c	Network Size
t	Image Frames Collections in Sequences
n	Number Of Body Joints,
$A^{\ell+1}$	Spatial Graph Convolution
$\widetilde{I^{(p)}}$	Adaptive Matrix Representation
$\Lambda_{x,x}^{(p)}$	Diagonal Matrix Representation
P	Probability Map
w^{ℓ}	Weight trajectory for the provided input.
δ	ReLU activation mechanism
$\otimes$	element-wise multiplication function
m	trainable weight matrix
$\mathcal{R}$	Graph depictions
f	Feature map
m_{ST}	Spatial and temporal attention maps
g	multi-scale graph convolution function
$\|\|$	Concatenate function
w_c	channel weight
m_z	Heat Maps
Z	Location of Key-Points
Y	Spatial Cue Modules
N	Visual Cue Modules
ϑ	Spatial Cue parameters
u_{ℓ}, v_{ℓ}	input and output pairs in ℓ
$[.]$	Integrating operation
n	Number Of Cues
c/n	Number of resultant channels
$Z_z^{c/n}$	Temporal Convolution Kernel
f_N	Intra-cue feature sequences
U	Inter-cue feature sequences
h_0	Initial state
h_t	Concealed state
$\mathcal{g}$	Sign Gloss Sequences
π	Alignment Path
$\mathcal{C}_v$	Context Vector
d_v	Division Channel
S	Probabilistic Sentences Chance
$\mathcal{L}_p$	Loss of pose estimations
$\mathcal{L}_{\xi}$	Pre-Training Loss

human pose and can focus on instructive joints and skeleton patterns.

The DDSTGCNN uses skeleton data to create higher-level feature mappings and joint positional trajectory on graph nodes. It employs GCNN techniques to explore joint spatial properties and temporal convolution processes to discover correlations. The network consists of 10 blocks, each with a DGCNN and DSTCNM for extracting features. These features are then used in SA-LSTM and CTC decoders for sequence learning and prediction using the BLSTM encoder.

3.2.1. Dense graph convolution neural network (DGCNN)

Dense graph convolution function (DGCF) is used to extract the skeleton data as a graph $g(v,e)$, where v and e is the collection of body joints and bones. Furthermore, temporal edges connect identical joints in two following image frames. After the graph has been obtained, temporal convolution functions are employed to identify temporal links by first exploring spatial properties between joints using graph convolution operations. Thus, consider $A^{l} \in R^{c*t*n}$ and $A^{l+1} \in R^{c*t*n}$ will be the depictions preceding and following the ℓ^{th} graph convolution where c, t, and n are the network size, image frames collections in sequences and number of body joints, correspondingly.

Define $A \in \{0,1\}^{n*n}$ is the skeleton graph's neighbouring matrix. $A_{x,y} = 1$ if the x^{th} and y^{th} joints are integrated, otherwise it is 0. The spatial graph convolution is denoted by,

$$A^{\ell+1} = \sum_{p\in P} w^{\ell} A^{\ell} (I^{\widetilde{(p)}}) \quad (1)$$

Here, $I^{\widetilde{(p)}} = \Lambda^{(p)^{-1/2}} I^{(p)} \Lambda^{(p)^{-1/2}} \in \mathcal{R}^{n*n}$ and $\Lambda_{x,x}^{(p)} = \sum_{p\in P} \Lambda_{x,y}^{(p)} + \gamma$ is the standardized adaptive matrix and diagonal matrix respectively. γ is set to 0.001 to eliminate empty rows I. w^{ℓ} is the weighting procedure that returns a weight trajectory for the provided input. P is the partitioned-set defined in the spatio-temporal GCN (ST-GCN) as $P = root, centripetal, centrifugal\}$. I is therefore composed of $I^{(root)}$, $I^{(cebtripetal)}$ and $\sum_{p\in P} I^{(p)} = I$. These spatial GCN algorithms integrate the features of every joint and its neighbours. Eq. (1) generates GCNs by combining joint depictions, determining 1-distance neighbors' local structural data, but convolution steps reduce local information, affecting prediction outcomes. The DGCNN operation leverages prompt associations to train the projection between the content of the former and hop joints which is indicated below.

$$A^{\ell+1} = \delta(\sum_{p\in P} w^{\ell} A^{\ell} (m^{(p)} \otimes I^{\widetilde{(p)}} + A^{\ell}) \quad (2)$$

Where, $m \in R^{n*n}$ is a trainable weight for every division category. A trainable weight matrix m is combined with an adjacent matrix, enabling the

proximity matrix to train autonomously across the network. The exact weights investigate joint motions, while shortcut connections increase the receptive field for local structural data.

Spatio-temporal attention network (STAN) focuses on vital frame and data. Flexible weight integrative method unifies spatial features which prevents the interference, enables GCNN operation to automatically acquire discriminate spatial features improving identification accuracy. The residual graph convolution approach uses an additional feature map $f = \mathcal{R}^{c*t*n}$ as input, combining spatial input with spatial dimension using 1D average pooling. Multi-layer perception models temporal associations using temporal attention map $f = \mathcal{R}^{c*t*n}$, utilizing 1D average pooling and activation mechanism to formulate spatial attention map $m_S \in \mathcal{R}^{c*t*1}$. The spatial and temporal attention maps are determined as $m_{ST} \in \mathcal{R}^{c*t*n}$, and stated below.

$$m_t(f) = sigmoid(MLP(avgpool\ (f))) \quad (3)$$

$$m_{ST}(f) = m_T^t \otimes m_S \quad (4)$$

The input feature map is stacked with an attention mask $m_{ST} \in \mathcal{R}^{n*c*t}$ for adaptive feature refining, utilizing spatial structural data and temporal dynamics. Attention modules pool data, broaden network receptive area, and detect previously unobserved connections. Enhancing residual branches is crucial for multi-scale data integration. The DGCNM can be expressed as,

$$A^{\ell+1} = \delta(\sum_{p \in P} w^{\ell} A^{\ell}(m^{(p)} \otimes \widetilde{I^{(p)}}) + m_{ST} \otimes A^{\ell}) \quad (5)$$

The GCNN's layer-by-layer restrictions enables the effective hidden connections between graph nodes and edges using transitional features. Then, the Eq. (5) is modified as follows,

$$A^{\ell+1} = f\left(g\left(A^{\ell}, w^{\ell}\right), A^{\ell}\right) = f(g\left(A^{\ell}, w^{\ell}\right), .., g(A^0, w^0), A^0) \quad (6)$$

The multi-scale graph convolution function is expressed by g, whereas the residual mapping operation is indicated by f. As the number of graph convolutional layers grows, the network is able to capture full spatial-temporal details from the sparse input by incorporating and distributing low- and high-level features throughout the graph.

3.2.2. Dynamic spatial-temporal convolution network module (DSTCNM)

The temporal properties of skeleton patterns are depicted in 1D convolution. The temporal convolution operation for graphs can generalize temporal features but can be challenging for visual classification tasks. DSTCNM, addresses these problems by continuously selecting the features from different temporal convolution kernel modules using channels weights. The module uses three convolution kernels sizes of 9*1, 5*1, and 3*1 to integrate temporal data across multiple scales. To ensure the network, the appropriate temporal receptive field range with three features are connected simultaneously in the channel weights and interactively choose the temporal features at various indications in relation to the channel weight. The DSTCNM is formulated as:

$$A_T^{\ell+1} = w_c(A_T^{\ell}(z_T = 9)||A_T^{\ell}(z_T = 5)||A_T^{\ell}(z_T = 3)) \quad (7)$$

Where, || represents the concate function. The channel weight w_c is used to modify the representation of temporal features at distinct scales. The model might continuously learn differentiating temporal properties and constantly change the temporal convolution receptive field using this technique of dynamic weight arrangements.

In this method, the dense connection is employed to interconnect the layers in each block. The complete network topology consists of ten blocks including a DGCF and a DSTCNM. Each block has 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 128, 128, 128, 256, 256, and 256 output channels. The initial stage involves the introduction of a data batch normalisation (BN) layer to normalise the incoming data. The feature maps of several identical-sized samples are combined in the subsequent evaluation using a global average pooling layer.

3.3 VGG11+1D-CNN+BLSTM feature extractor for full frame data

The spatial representation module employs 1D-CNN to construct multi-cue features from full frame data by utilizing the VGG11 model as the backbone network. It performs posture prediction, patch cropping, and feature extraction using full frame data.

3.3.1. Pose valuation

Deconvolutional networks are commonly used for pixel-wise detection. In the VGG11 model, two de-convolutional layers are introduced for posture

determination, with each layer having a 2 stride. The feature maps are up-sampled by a factor of 4 which is fed into a point-wise convolutional layer and projected D heat maps as output with the position of matching key-points having the maximum response value. A softmax layer is applied to these heat maps for differentiating important key-point like nose, chin, left and right side shoulders, elbows and writs predictions in sequence training. Indicating D heat maps as $m = \{m_z\}_{z=1}^{Z}$, where each heat maps $m_z \in \mathcal{R}^{H*W}$ is subjected to the following spatial softmax function as

$$P_{x,y,d} = \frac{j^{m_{x,y,d}}}{\sum_{x=1}^{H}\sum_{y=1}^{W} j^{m_{x,y,d}}} \quad (8)$$

In Eq. (8), $j^{m_{x,y,d}}$ represents the value of the heat map m_z at location (x, y) and $P_{x,y,d}$ represents the probability of key-points z at position (x, y). The expected results of coordinates across x and y-axes of a probability map are then established as follows:

$$(\hat{\imath}, \hat{\jmath})_z = \left(\sum_{x=1}^{H}\sum_{y=1}^{W}\frac{x-1}{H-1}\, P_{x,y,d}, \sum_{x=1}^{H}\sum_{y=1}^{W}\frac{y-1}{W-1}\, P_{x,y,d}\right) \quad (9)$$

In the preceding Eq. (9), $Y_z = (\hat{\imath}, \hat{\jmath})_z \in [0,1]$ is the normalised expected location of key-points z. In a $H * W$ feature map, the equivalent location of $(\hat{\imath} * (H-1)+1, \hat{\jmath}\,(W-1)+1)$.

3.3.2. Patch cropping

It is important to evaluate different visual cues like lip expression, facial expression, form, hand shape, and hand movements. The model uses nose and wrist locations as face and hand center coordinates. Patches are created using VGG-11 network's fourth convolutional layer, cropping hands at $24 * 24$ and faces at $16 * 16$. Cameras can see the signer's upper body due to its size. The patche' centers are kept within a specified range to avoid overlap between fake and actual feature maps.

3.3.3. Feature generation

The position cues feature vector is obtained by compressing predicted Z key-points into a 1D vector with dimensions $2Z$ and running it through two fully connected (FC) layers using ReLU. The face and hands feature maps are processed individually using convolutional layers, with both hands being used for most sign movements. The merged results are then merged along channel variables. The global average pooling of hands, face, and full-frame feature mappings is then combined to create feature vectors of the whole frame of $B * 7 * 7 * 512\ B * 512$. All features are retrieved by feeding frames $i = \{i_b\}_{b=1}^{B}$ through the spatial cue module, which is described below.

$$\left\{\{f_{b,N}\}_{N=1}^{n}, \{Y_{b,z}\}_{N=1}^{n}\right\}_{b=1}^{B} = \{\Omega_\vartheta\,(i_b)_{b=1}^{B}\} \quad (10)$$

Where, the location of key point z at the f^{th} frame is represented by $Y_{b,z} \in R^2$, indicates the spatial cue module with parameters ϑ. The suggested temporal cue module incorporates intra-cue and inter-cue spatial-temporal information, with the inter-cue pathway incorporating integrated elements from multiple cues occurring at varying times. A temporal module replicates the interpretation between these pathways which is given below.

$$(u_\ell, v_\ell) = Block_\ell(u_{\ell-1}, v_{\ell-1}) \quad (11)$$

Where, $(u_{\ell-1}, v_{\ell-1})$ and (u_ℓ, v_ℓ) are the ℓ^{th} block's input and output pairs. The feature matrix of the inter- and intra-cue is denoted by $u_\ell \in R^{B*c_0}$ and $v_\ell \in R^{B*c_f}$ which is a fusion of vectors from distinct cues as well as channel-dimensions. $u_1 = f_1 = [f_{1,1}, f_{1,2}, \ldots., f_{1,n}]$, where $[.]$ is the integrating operation and n is the number of cues.

Intra-cue path

An initial route establishes the defining features of numerous inputs at various time scales. Each cue undergoes a temporal modification as described below,

$$f_{\ell,n} = ReLU\,(\mathcal{Z}_z^{c/n}\,(f_{\ell-1,\ N})) \quad (12)$$

$$f_\ell = [f_{1,1}, f_{1,2}, \ldots., f_{1,n}] \quad (13)$$

In the preceding Eqs. (12) and (13), $f_{\ell,n} \in R^{B*c/n}$ denotes the N^{th} cue's feature matrix, and $\mathcal{Z}_z^{c/n}$ depicts the temporal convolution kernel and c/n is the number of resultant channels.

Inter-cue path

The intra-cue path data is coupled with the second-path data to determine the changes in the prior block's inter-cue properties over time.

$$u_\ell = ReLU\,([\,\mathcal{Z}_z^{c/2}\,(u_{\ell-1})\mathcal{Z}_z^{c/2}\,(f_\ell)]) \quad (14)$$

A point-wise temporal convolution is $\mathcal{Z}_1^{c/2}$ representing two pathways. The output of the

intra-cue path is f_{ℓ}. A temporal max-pooling with stride 2 and kernel size 2 is run after each block. The system uses two blocks, with all temporal convolutions having a kernel size of 5. Each path has 256 output channels c with a value of 1.

3.4 Sequence learning

The network may produce intra-cue feature sequences $f_N = \{f_{t,N}\}_{b=1}^{B'}$ and inter-cue feature sequences $U = \{u_b\}_{b=1}^{B'}$ using the suggested DDSTGCNN and VGG11+1D-CNN+BLSTM module. The temporal length of the temporal cue module's output in this case is B'. The challenging aspect is expressing the reliant possibilities $(\mathcal{g}|u, f)$ and $\mathcal{p}(w|u, f)$ using these two feature patterns for sequence learning.

BLSTM encoder: Recurrent neural networks (RNN) simulate the state modifications of input sequences by adjusting their internal state. RNN transform spatial-temporal sequences into sign-gloss sequences and produce B' hidden states by reading feature sequences as input.

$$h_t = RNN_{jN}(h_{t-1},\ u_b) \tag{15}$$

The initial state h_0 is a constant all-zero vector at time t, while the concealed state h_t is determined by Eq. (15). The BLSTM unit is used for its superior performance in handling long-term dependencies. It process the data from both directions simultaneously and the FC and softmax layers transmit the concealed state for each time step which is stated below.

$$Q_b = w_j h_t + r_j\ , \mathcal{p}_{t,y} = \frac{j^{V_{b,y}}}{\sum_z j^{V_{b,y}}} \tag{16}$$

Here, the chance of gloss y at time t is denoted by $j^{V_{b,y}}$. The gloss y used in ISL recognition tests is drawn from a predetermined corpus of words.

CTC decoder for ISL recognition: The proposed approach utilizes the CTC decoder to map the unlabeled video patterns $U = \{u_b\}_{b=1}^{B'}$ to arranged sign gloss sequences $\{\mathcal{g}_a\}_{a=1}^{A'}(A \le B')$. CTC increases the probability of correct alignments between source and target sequences. An enlarged vocabulary $\mathcal{V}$ is labelled with a blank " $-$ " which is derived as $= \mathcal{V}_{origin} \cup \{-\}$. The blank represents fluidity and irrelevance in the transition. The input sequences are aligned according to the route $\pi_b = \mathcal{V}$ which is labelled as $\pi = \pi_{b=1}^{B'}$ illustrating the possibility of an alignment path π for the given input sequences which is depicted in Eq. (17).

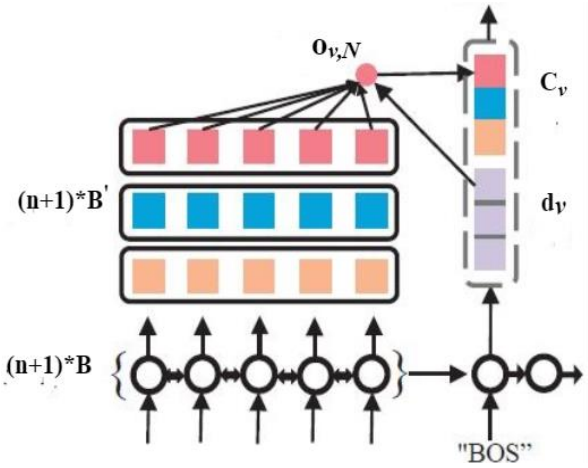

Figure. 2 Illustration of the SA mechanism

$$p(\pi|u) = \prod_{b=1}^{B'} p(\pi_b|u) = \prod_{b=1}^{B'} j_b, \pi_b \tag{17}$$

Eq. (17) describes a many-to-one mapping function b that removes interruption and repeated words from alignment routes.

To determine conditional chances, the odds of all paths linked to sign gloss patterns $\mathcal{g}$ through $\mathcal{b}$ are added.

$$p(\mathcal{g}|u) = \sum_{\pi \in \mathcal{b}^{-1}(\mathcal{g})} p(\pi|u) \tag{18}$$

Where, the reverse function of $\mathcal{b}$ is represented by $\mathcal{b}^{-1}(\mathcal{g}) = \{\pi|\mathcal{b}(\pi) = \mathcal{g}\}$. The aforementioned expression applies to the conditional probability $p(\mathcal{g}|f_N)$ based on the intra-cue.

SA-LSTM decoder for translation: A series of hidden state sequences $\mathcal{H} = (h_1, h_2, \dots, h_{n+1}) = \left\{\left\{h_{N,b}{}_{b=1}^{B'}\right\}\right\}_{N=1}^{n+1}$ is produced by the BLSTM encoder. Here, $h_1 = R^{B'*c_0}$ and $h_N = R^{B'*c_f}$ $(N > 1)$ are derived from the inner-cue and intra-cue, respectively. The spoken language translation decoding process is described as follows:

$$d_v = RNN_{ED}(d_{v-1}, w_{v-1}) \tag{19}$$

The text describes a unidirectional LSTM recurrent unit with hidden paths d_v and w_{v-1} for forecasting word tokens. Fig. 2 depicts the segmented attention (SA) method is shown for modifying it to a multi-cue framework.

The state S_0 is initialized by a linear layer in conjunction of $\{h_N, B'\}_{N=1}^{n+1}$ and autoregressive tasks initiates the word tokens for predicting beginning of a sentences (BOS) and end of a sentences (EOS). The conditional probability $p(w_v|\mathcal{H})$ is computed at each decoding level to maximize information sources. The attention mechanism provides contextual data and

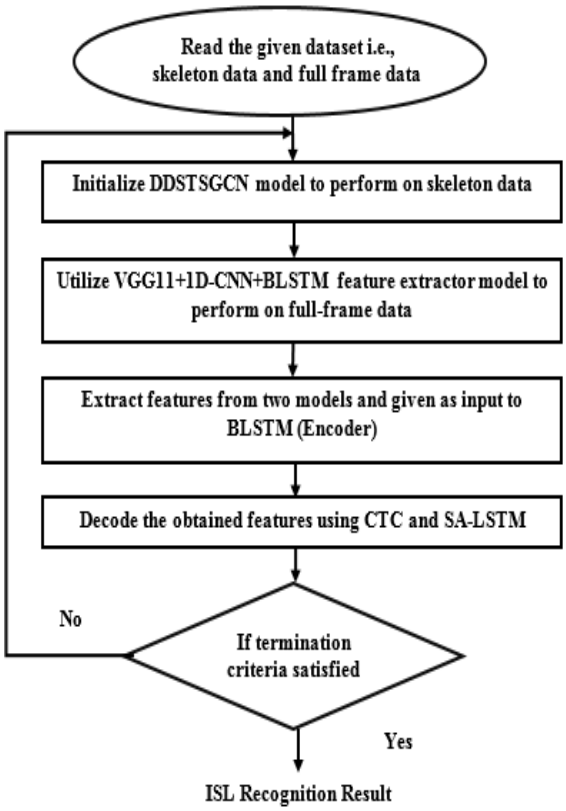

Figure. 3 Flowchart for the proposed model

directs the decoder network's attention to a specific subset of the encoder's output. The context vector $\mathcal{C}_v$ is computed using Long's approach, a widely used technique in neural machine translation (NMT).

$$\mathcal{H} = [h_1, h_2, \dots, h_{n+1}] \tag{20}$$

$$o_{v,N} = SOFTMAX(d_v\, w\mathcal{H}^T) \tag{21}$$

$$\mathcal{C}_v = o_v\, \mathcal{H} \tag{22}$$

The concatenation values and channel axis are represented as $\mathcal{H} \in R^{B'} * nc_f$. Uniform attention weights $o_v \in R^{B'}$ are required, making normal attention ineffective. The attention weights are assigned for each cue individually using a Single BLSTM encoder for inter-cue features and n for intra-cue features. The SA-LSTM decoder output is divided along channels for context vectors from multiple cue sources. The $\mathcal{C}_v$ is computed as follows,

$$d_v = [d_{v,1}, d_{v,2}, , \dots . , d_{v,n+1}] \tag{23}$$

$$o_{v,N} = SOFTMAX(d_{v,N}\, w_N\, h_N^T) \tag{24}$$

$$\mathcal{C}_v = [d_{v,1}h_1, d_{v,2}h_2, \dots . , d_{v,n+1}h_{n+1}\,] \tag{25}$$

Where, d_v is evenly divided into $n+1$ segments along the channel axis. By integrating d_v and $\mathcal{C}_v$, the chance distribution of words at v^{th} is created.

$$\overbrace{d_v} = \tanh(w_d\,[d_v, \mathcal{C}_v] + r_d\,) \tag{26}$$

$$k_v = softmax(w_d \overbrace{d_v} + \overbrace{d_\varrho}\,) \tag{27}$$

The probabilistic chance of the sentence S is stated as follows via word-by-word translation:

$$p(S|u,f) = \prod_{v=1}^{V} p\,(S_v|S_{v-1}, \mathcal{H}) = \prod_{v=1}^{V} k_v, S_v \tag{28}$$

3.5 Inference

The optimization of ISL recognition and translation is persistent with demonstrating efficiency of STHCN through a two-stage approach. The Fig. 3 represents the flowchart of this methodology.

ISL recognition

The inter-cue and intra-cue sequence losses are specified as u and f_N in the CTC decoder as follows,

$$\mathcal{L}_{CTC-u} = -lnp(g|u) \tag{29}$$

$$\mathcal{L}_{CTC-f_N} = -lnp(g|f_N) \tag{30}$$

The essential target of training should be the enhancement of the inter-cue route. The intra-cue route serves as an auxiliary to the fusion process which provides the data about each single cue.

As a result, the joint optimisation loss (JL) for ISL recognition is represented by the following expression:

$$\mathcal{L}_\xi = \mathcal{L}_{CTC-u} + \alpha \sum_N \mathcal{L}_{CTC-f_N} + \mathcal{L}_p^{[\beta]} \tag{31}$$

The hyper-parameter controls auxiliary loss and regression loss $\mathcal{L}_p$ in pose estimations. It computes smooth- l_1 loss operation using estimated keypoints $Y_{b,z} \in \mathcal{R}^2$ wi with appropriate ground-truth $\hat{Y}$ is computed as follows,

$$\mathcal{L}_p^{[\beta]} = \frac{1}{2BZ} \sum_b \sum_z \sum_{x\in(i,j)} smooth_{l_1}\beta(Y_{b,z,x} - \hat{Y}_{b,z,x}) \tag{32}$$

in which,

$$smooth_{l_1}(i) = \begin{cases} 0.5i^2 & if|i| < 1 \\ |i| - 0.5 & otherwise \end{cases} \tag{33}$$

The fast-RCNN generates smooth- l_1 loss for bounding box regression learning, while pseudo-

(a) (b) (c) (d)

Figure 4 Sample mapped annotation from the video frame images: (a) Frame (video image), (b) Annotations, (c) Frame (video image), and (d) Annotations

label-based pre-training improves model depiction, consistency, and reduces outliers.

Eq. (34) calculates pre-training loss.

$$\mathcal{L}_{\xi} = \mathcal{L}_{CE-u} + \alpha \sum_N \mathcal{L}_{CE-f_N} + \mathcal{L}_{\mathcal{p}}^{[\beta]} \quad (34)$$

Where CE will be cross-entropy loss which is relevant to the identification task for each word in sentences.

ISL translation

The SA-LSTM decoder is attached to create language translations by utilizing the pre-trained models from ISL identification. The objective function of ISL translation is depicted in Eq. (35),

$$\mathcal{L}_T = -ln\mathcal{p}(w|u, f_N) \quad (35)$$

Video frames are routed through the DSTGCNN and VGG11+1D-CNN+BLSTM modules for inference. The inter-cue feature sequence and BLSTM encoder generates the subsequent probability distribution of sentences at complete time intervals. The SA-LSTM decoder auto-regressively integrates inter and intra-cue features to forecast ISL and phrases, maintaining distinctiveness of each cue and examining synergy among them.

Hence, this STHCN model effectively resolves the issues of time complexity issues manual annotation of key-points for ISL recognition and translation. The Fig. 4 represents the mapped annotation from the collected video frame images.

4. Result and discussion

4.1 Dataset description

For the experimental purposes, the dataset is gathered from [23]. The ISL-continuous sign language translation and recognition (ISL-CSLTR) dataset aims to improve the SLTR framework for interaction with deaf and dumb society using deep learning and computer vision techniques. The corpus includes 700 annotated movies, 18863 sentence-level frames, and 1036 word-level pictures from 100 spoken language sentences signed by seven signers. It is accessible to the public and organized according to signer variations and temporal restrictions. The corpus is annotated with relevant spoken language phrases for clear interpretation.

4.2 Performance analysis

In this section, the efficacy of the STHCN model is compared with existing models such as STMC [12], SIGNBERT [15], H-DNA [16], MOPGRU [19] and YOLOv4 [20] by executing them in Phyton. For the experimental study, the proposed and existing models are implemented and tested using the ISL-CSLTR dataset which is discussed in section 4.1. From the collected dataset, 610 videos are considered in which 60% (366) is used for training and the remaining 40% (244) for testing. The final output consists of 86 sentences and 25 frames per second. Table 2 lists parameter settings for the STHCN and the existing models for ISL recognition and translation.

The assessment measures used to determine the effectiveness of the proposed and existing models are depicted briefly below.

Accuracy is defined as the number of effectively identified signs is proportional to the total number of signs used for categorisation, indicating that the model was accurately trained and how it may perform in average.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (35)$$

Table 2. Parameter settings for existing and proposed model

Model	Parameters	Range
STMC [12]	Learning rate	5×10^{-5}
	Loss Function	Cross-Entropy
	Optimizer	SGD
	Batch size	24
	Hidden Units	1280
	Activation function	ReLU
SignBERT [15]	Learning rate	1 × 10−3
	Loss Function	Cross-Entropy
	Optimizer	Adam
	Weight Decay	5×10^{-5}
	Hidden Units	256
H-DNA [16]	Learning rate	0.01
	Loss Function	Cross Entropy
	Optimizer	Adam
	Hidden unit	256
	Activation function	Sigmoid and Tanh
MOPGRU [19]	Loss Function	Cross Entropy
	Optimizer	Softmax function
	Hidden unit	512
	Output Layer	36
	Total epochs	50
	Activation function	ReLU
YOLOv4 [20]	Learning rate	0.001
	Optimizer	Adam
	Decay	0.0005
	Filter size	128
	Batch size	64
	Activation function	Linear, ReLU
Proposed STHCN	Learning rate	0.01
	Loss Function	Cross Entropy
	Optimizer	SGD
	Embedding Size	512
	Hidden Layer	1024
	Output Channel	256
	Dropout rate	0.5
	Stride	2
	Filter size	32
	Batch size	64
	Activation function	ReLU

In Eq. (35), true positives (TP) indicate the model accurately predicting the sign as actual instances, true negatives (TN) indicate it predicts a sign that does not belong to the instances, false positives (FP) contradict the actual instances as false and projected instances as true, and false negatives (FN) show a model prediction that contradicts itself, as the actual sign instance is true while the detected class is false.

The precision score is employed in SLR to evaluate model performance as it primarily indicates about false positive results in the dataset. A higher accuracy score means fewer false positive values.

$$Precision = \frac{TP}{TP+FP} \tag{36}$$

The recall score is utilized to measure model efficiency in SLR since it mainly indicates about false negative values in the dataset. A better recall score indicates fewer false negatives.

$$Recall = \frac{TP}{TP+FN} \tag{37}$$

The Fig. 5, 6 and 7 portrays the accuracy, precision and recall values obtained by the proposed and existing models for predicting the ISL performance. From the fig 5, it is demonstrated that the accuracy of the STHCN model enhances by .87%, 5.45%, 4.33%, 3.74%, and 1.94% when comparing with the existing models SIGNBERT, STMC, H-DNA, MOPGRU, and YOLOv4 models respectively, Similarly, the precision values of STHCN significantly increased by 9.32%, 6.22%, 5.88%, 3.12%, and 1.80% compared to the existing algorithms as depicted in fig 6. In fig 7, the recall analysis of STHCN is maximized by 9.97%, 5.44%, 3.73%, 2.89%, and 1.43% when compared to the existing algorithms respectively.

From the above observations, it is analyzed that proposed STHCN algorithm achieves highest performance in terms of accuracy, precision and recall compared to other models due to the use of various features and annotations, which are learned automatically with lower effort which effectively improves the ISL recognition and translation.

Fig. 8 presents the time complexities of proposed and existing ISL recognition methods on the ISL-CSLTR dataset. It is noted that the proposed STHCN algorithm can efficiently minimize the time complexities of predicting ISL performance by considering all criteria's, compared to other existing algorithms. The STHCN algorithm decreases about 50.45%, 41.27%, 36.57%, 25.50% and 13.28% in contrast with the SIGNBERT, STMC, H-DNA, MOPGRU, and YOLOv4, respectively.

5. Conclusion

In this paper, STHCN model is developed to reduce time complexity and avoid manual keypoint annotations in ISL recognition and translation. It uses the DGCF to prevent early data loss of local nodes during graph convolution, while the STAN module distinguishes crucial frames and joints, minimizing data changes during dense connection operations. DSTCNM is constructed to provide a range of temporal graph receptive fields. VGG11+1D-

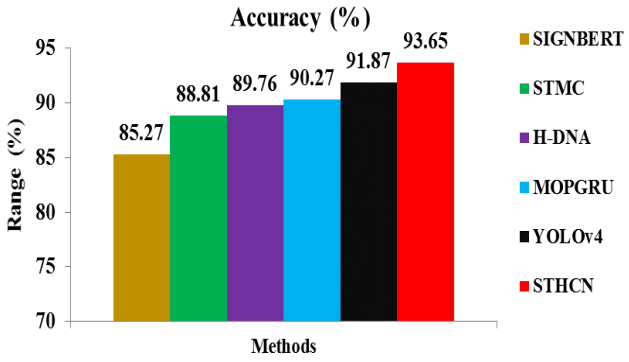

Figure. 5 Accuracy comparison for proposed and existing models

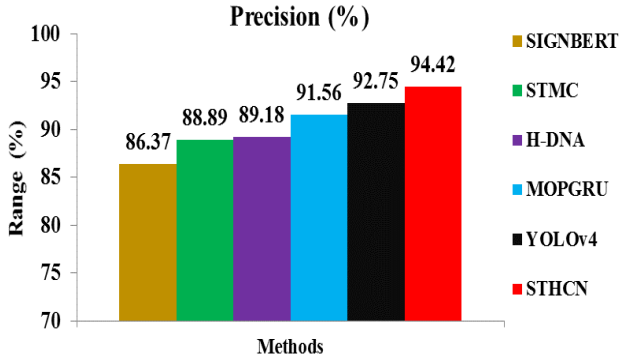

Figure. 6 Precision comparison for proposed and existing models

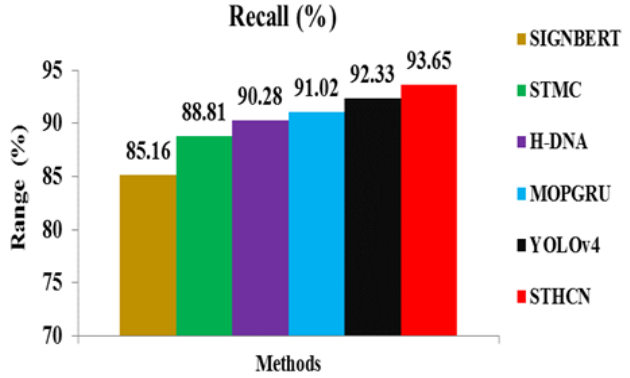

Figure. 7 Recall comparison for proposed and existing models

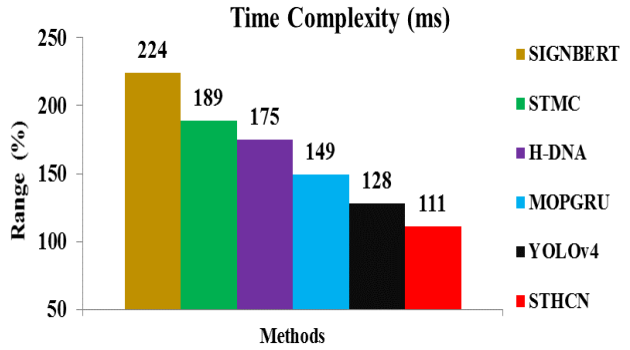

Figure. 8 Time complexities comparison for proposed and existing models

CNN+BLSTM is used for feature extraction, and the extracted features are modified using the BLSTM encoder. The SA-LSTM decoder performs auto-regressive forecasting on captured video frames for ISL detection. Extensive experiments show the STHCN model achieves 93.65% accuracy on the ISL-CSLTR in detecting ISL.

Conflict of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, methodology, software, validation, Prema; formal analysis, investigation, Gomathi; resources, data curation, writing—original draft preparation, Prema; writing—review and editing, Prema; visualization; supervision, Gomathi;

References

[1] D. Dhake, M. P. Kamble, S. S. Kumbhar, and S. M. Patil, "Sign language communication with dumb and deaf people", *Int. J. Eng. Appl. Sci. Technol.*, Vol. 5, No. 4, pp. 254-258, 2020.

[2] S. Suthagar, K. S. Tamilselvan, P. Balakumar, B. Rajalakshmi, and C. Roshini, "Translation of sign language for Deaf and Dumb people", *Int. J. Recent. Technol. Eng.*, Vol. 8, No. 5, pp. 4369-4372, 2020.

[3] O. Koller, J. Forster, and H. Ney, "Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers", *Computer Vision and Image Understanding*, Vol. 141, pp. 108-125, 2015.

[4] A. V. Nair and V. Bindu, "A review on Indian sign language recognition", *International Journal of Computer Applications*, Vol. 73, No. 22, 2013.

[5] B. Garcia and S. A. Viesca, "Real-time American sign language recognition with convolutional neural networks", *Convolutional Neural Networks for Visual Recognition*, Vol. 2, No. 8, pp. 225-232, 2016.

[6] R. Rastgoo, K. Kiani, and S. Escalera, "Sign language recognition: A deep survey", *Expert Systems with Applications*, Vol. 164, p. 113794, 2021.

[7] I. A. Adeyanju, O. O. Bello, and M. A. Adegboye, "Machine learning methods for sign language recognition: A critical review and

analysis", *Intelligent Systems with Applications*, Vol. 12, p. 200056, 2021.

[8] P. Jayanthi, P. R. Bhama, K. Swetha, and S. A. Subash, "Real Time Static and Dynamic Sign Language Recognition using Deep Learning", *Journal of Scientific & Industrial Research*, Vol. 81, No. 11, pp. 1186-1194, 2022.

[9] N. Aloysius and M. Geetha, "Understanding vision-based continuous sign language recognition", *Multimedia Tools and Applications*, Vol. 79, pp. 22177–22209, 2020.

[10] R. Dhiman, G. Joshi, and C. R. Krishna, "A deep learning approach for Indian sign language gestures classification with different backgrounds", *Journal of Physics: Conference Series*, Vol. 1950, No. 1, p. 012020, IOP Publishing, 2021, August.

[11] M. A. Hammadi, G. Muhammad, W. Abdul, M. Alsulaiman, M. A. Bencherif, T. S. Alrayes, and M. A. Mekhtiche, "Deep learning-based approach for sign language gesture recognition with efficient hand gesture representation", *IEEE Access*, Vol. 8, pp. 192527-192542, 2020.

[12] H. Zhou, W. Zhou, Y. Zhou, and H. Li, "Spatial-temporal multi-cue network for sign language recognition and translation", *IEEE Transactions on Multimedia*, Vol. 24, pp. 768-779, 2021.

[13] A. Mittal, P. Kumar, P. P. Roy, R. Balasubramanian, and B. B. Chaudhuri, "A modified LSTM model for continuous sign language recognition using leap motion", *IEEE Sensors Journal*, Vol. 19, No. 16, pp. 7056-7063, 2019.

[14] I. Papastratis, K. Dimitropoulos, D. Konstantinidis, and P. Daras, "Continuous sign language recognition through cross-modal alignment of video and text embeddings in a joint-latent space", *IEEE Access*, Vol. 8, pp. 91170-91180, 2020.

[15] Z. Zhou, V. W. Tam and E. Y. Lam, "SIGNBERT: a Bert-based deep learning framework for continuous sign language recognition", *IEEE Access*, Vol. 9, pp. 161669-161682, 2021.

[16] B. Natarajan, E. Rajalakshmi, R. Elakkiya, K. Kotecha, A. Abraham, L. Gabralla, and V. Subramaniyaswamy, "Development of an End-to-End Deep Learning Framework for Sign Language Recognition, Translation, and Video Generation", *IEEE Access*, Vol. 10, pp. 104358-104374, 2022.

[17] S. Katoch, V. Singh, and U. S. Tiwar, "Indian Sign Language recognition system using SURF with SVM and CNN", *Array*, Vol. 14, pp. 1-9, 2022.

[18] D. Kothadiya, C. Bhatt, K. Sapariya, K. Patel, A. B. G. González, and J. M. Corchado, "Deepsign: Sign language detection and recognition using deep learning", *Electronics*, Vol. 11, No. 11, p. 1780, 2022.

[19] B. Subramanian, B. Olimov, S. M. Naik, S. Kim, K. H. Park, and J. Kim, "An integrated mediapipe-optimized GRU model for Indian sign language recognition", *Scientific Reports*, Vol. 12, No. 1, pp. 1-16, 2022.

[20] R. Sreemathy, M. P. Turuk, S. Chaudhary, K. Lavate, A. Ushire, and S. Khurana, "Continuous word level sign language recognition using an expert system based on machine learning", *International Journal of Cognitive Computing in Engineering*, Vol. 4, pp. 170-178, 2023.

[21] Z. Cao, T. Simon, S. E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields", In: *Proc of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7291-7299, 2017.

[22] D. Li, "Human skeleton detection and extraction in dance video based on PSO-enabled LSTM neural network", *Computational Intelligence and Neuroscience*, 2021.

[23] https://data.mendeley.com/datasets/kcmpdxky7p/1#:~:text=The%20ISLCSLTR%20corpus%20consists%20of%20a%20large%20vocabulary,annotated%20details%20and%20it%20is%20made%20available%20publicly.