Spatiotemporal Attention-based Hybrid CNN-Transformer Model for Indian Sign Language Recognition

R Moushmi Srimayi
Dept. of Computer Science
CMR Institute of Technology
Bengaluru, India
mor22cs@cmrit.ac.in

K Umme Habeeba
Dept. of Computer Science
CMR Institute of Technology
Bengaluru, India
kuha22cs@cmrit.ac.in

Monika S V
Dept. of Computer Science
CMR Institute of Technology
Bengaluru, India
mosv22cs@cmrit.ac.in

Dr. Prem Kumar Ramesh
Dept. of Computer Science
CMR Institute of Technology
Bengaluru, India
premkumar.r@cmrit.ac.in

***Abstract*—Indian Sign Language (ISL) is the primary means of communication for the Deaf community, yet interaction with non-signers often requires human interpreters, which is not always practical or accessible. Existing research has primarily focused on isolated sign recognition (ISR), with limited progress toward complete end-to-end bidirectional translation frameworks. Continuous Sign Language Recognition (CSLR) remains challenging due to issues such as low accuracy, class imbalance, high memory requirements, and recurrent architectures affected by vanishing gradients. To overcome these limitations, we present DeepISL, an end-to-end system for real time ISL to English translation. For recognition DeepISL employs MediaPipe-based keypoint extraction and a Custom Hybrid CNN-Transformer with spatiotemporal attention, achieving 96.05% accuracy.A rule-based NLP grammar engine further refines raw gloss outputs into fluent English sentences.For text to ISL applies rule-based grammar reordering, gloss mapping, and skeletal animation synthesis using pre-recorded keypoints. Together, these components form a robust, low-latency, and linguistically accurate bidirectional ISL translation framework that bridges the communication gap between signers and non-signers.**

***Index Terms*—Indian Sign Language, Isolated sign recognition, Continuous Sign Language Recognition, CNN-Transformer, MediaPipe, Spatiotemporal Attention.**

I. Introduction

Effective communication is the cornerstone of societal integration and human connection. However, a significant communication gap persists between the hearing majority and the millions of individuals in the deaf and hard-of-hearing community, for whom sign language is a primary means of expression. In India, Indian Sign Language (ISL) serves this vital role, yet the scarcity of qualified interpreters creates substantial barriers in daily life, including education, healthcare, and public services. The development of automated translation systems offers a promising avenue to bridge this divide, empowering the deaf community with greater autonomy and fostering a more inclusive society. This research focuses on the design and implementation of a real-time, bidirectional system to translate between ISL and English text.

The primary challenge in this domain is creating a system that is both accurate and accessible. Existing solutions often fall short. Traditional approaches relying on human interpreters are not scalable and are often unavailable or prohibitively expensive. Early technological solutions frequently depended on specialized hardware like sensor-equipped gloves or complex multi-camera setups, limiting their practical, everyday use. Furthermore, many software-based systems are restricted to recognizing only static alphabets or a small vocabulary of isolated signs, failing to interpret the fluid, continuous, and context-dependent nature of conversational sign language. The complexity of capturing nuanced hand gestures, facial expressions, and body language in real-time from a standard video stream has remained a significant technological hurdle.

To address these limitations, we propose DeepISL, a novel, web-based framework for real-time, bidirectional Indian Sign Language translation. The primary objective of this work is to develop an accessible system that functions using only a standard webcam, eliminating the need for specialized hardware. Our system leverages the MediaPipe Holistic framework to simultaneously track 144 keypoints across the hands, face, and body pose from a live video feed. This holistic data is then processed by a deep learning model on a backend server to recognize continuous ISL gestures and translate them into English text. The novelty of our approach lies in its bidirectional functionality: not only does it translate sign language to text, but it also translates user-typed text into corresponding ISL video animations, creating a complete communication loop. This dual-capability, combined with its real-time performance and hardware accessibility, represents a significant advancement toward a practical and widely deployable sign language translation tool.

The remainder of this paper is organized as follows: Section II reviews related work in ISL recognition and generation. Section III details the proposed methodology, covering data collection, preprocessing, and model architecture. Section IV presents the results, comparative analysis, and discussion. Finally, Section V concludes the work.

II. Related Work

Recent research on sign language recognition and translation has looked into different ways to address challenges in the field. Early studies focused on hand-crafted features and mixed models. For example, Katoch et al. [1] used SURF features with SVM and CNN to improve the accuracy of Indian Sign Language (ISL) recognition. Areeb et al. [2] proposed a deep learning method to assist hearing-impaired individuals during emergencies. This demonstrated the real-world potential of sign language recognition systems.Today, deep learning methods are at the forefront of research, with new architectures being developed. Natarajan et al. [3] created an end-to-end framework for recognition, translation, and video generation. Their work showed the benefits of unified processes. Sharma and Singh [4] introduced a spatio-temporal framework for continuous ISL recognition, which captured temporal dependencies to manage ongoing gestures. Khartheesvar et al. [5] combined MediaPipe Holistic with LSTM networks for automatic recognition, focusing on lightweight and real-time execution. Hon et al. [6] also used CNNs for real-time ISL recognition, confirming that convolutional architectures can be effective for speed-sensitive applications.Other researchers have investigated innovative architectures. Ghorai et al. [7] proposed a system that utilized network deconvolution and spatial transformer networks. In a subsequent study [8], they introduced TSI-CNN-Net, a shift-invariant CNN that improved recognition in changing spatial conditions. Das et al. [9] developed a deep ISL recognition framework and further improved it in [10] to handle occlusion challenges using CNN and pose features. Bansal and Jain [11] suggested a dual feature descriptor with GMT-MASKRCNN, showing strong results in video-based recognition.Generative techniques have also been studied. Sreemathy et al. [12], [13] used Generative Adversarial Networks (GANs) for producing video from text. This enabled realistic avatar-based communication. These studies highlight how generative AI can enhance datasets and improve human-computer interaction. Najib [14] applied AI-powered interpretation methods to boost accessibility through machine learning-driven sign-to-text translation. Survey and review studies have played an important role in the field. Sabharwal and Singla [15] provided a thorough review of ISL translation to text. They described existing methods, challenges, and potential future directions. Their work emphasized the need for scalable and robust models that can accommodate variations among signers, diverse vocabularies, and complex sentence-level recognition.In summary, these contributions show significant advancements in ISL recognition and translation. They tackle hand-crafted methods, deep learning architectures, generative models, and mixed approaches. Nevertheless, challenges persist in achieving signer independence, working with limited datasets, ensuring real-time processing in uncontrolled settings, and delivering precise sentence-level translations. Future research will continue to focus on these areas.

III. Methodology

Our system utilizes a custom CNN-Transformer hybrid architecture for sign language recognition (SLR). While CNNs are highly effective at extracting spatial features from individual frames, they are limited in capturing long-range temporal dependencies, which are essential for recognizing dynamic signs. Traditional methods that combine CNNs with recurrent networks, such as LSTMs, often encounter several challenges, including vanishing gradients, sequential processing bottlenecks, slower training, and limited capacity to model global temporal context. To overcome these limitations, our architecture replaces the recurrent module with a Transformer encoder. The model employs a CNN front-end for spatial feature extraction, followed by a Transformer module that leverages self-attention to model the entire temporal sequence in parallel. This design provides a more powerful and efficient framework for temporal modeling compared to conventional CNN-RNN approaches. The proposed system is structured as a bidirectional translation framework, comprising two key modules: (A) ISL-to-Text, which converts sign language sequences into textual representation. (B) Text-to-ISL, which generates sign language sequences from textual input.

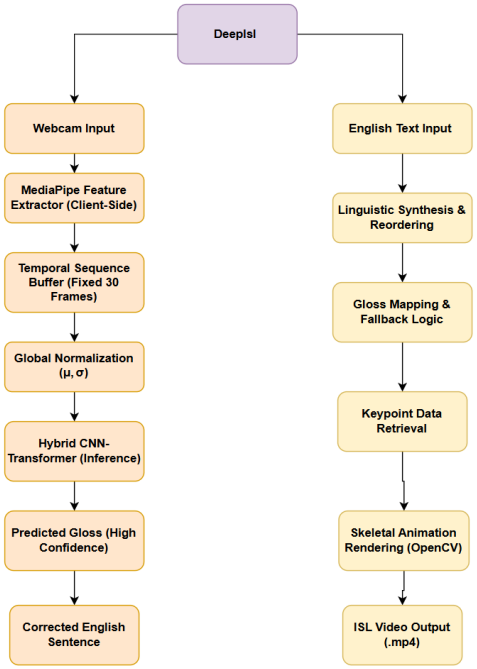

Fig. 1: DeepIsl, bidirectional Translation System flow chart

A. ISL-to-Text

1) Data Collection: Lack of large public, standardized datasets for Indian Sign Language (ISL) Sign Language Recognition (SLR) systems critically hinders the development

of strong ISL SLR systems [1, 9]. It compels researchers to prepare bespoke datasets that are limited in diversity, realism, and scale. Most current ISL resources are piecewise, greatly biased toward isolated signs or static alphabets, and are based on raw video storage, which is computationally expensive and has high frame-level redundancy [2, 5]. Such video data processing requires heavy preprocessing, such as keyframe extraction and background subtraction, and the resulting models have poor generalization because of high susceptibility to environmental changes (e.g., lighting, background clutter) and signer-specific features [7, 10]. To address these constraints, we created a new, lightweight dataset designed for real-time SLR. Following the contemporary move towards skeleton representation [3, 5], our data collection pipeline leverages MediaPipe Holistic to obtain 2D skeletal keypoints in real-time during recording without storing the raw video at all. This method leads to a small and computationally cheap dataset in which every sign is encoded as a clean, 30-frame sequence of landmark coordinates, free of visual noise by nature and suitable for model input. Our corpus includes 38 different ISL signs, a carefully curated vocabulary that covers the full manual alphabet and a list of high-frequency words, thus enabling spelling-based as well as whole-word communication. Recognizing the usual limitation of low signer involvement [6, 8], we deliberately imposed controlled intra-sample variations such as lighting changes, camera angle, and signing speed on the corpus to artificially augment the dataset's diversity and resilience. This intentional introduction of variability assists in reducing overfitting and enhances the generalizability of the model to unknown users and settings [10]. This approach of coupling an effective keypoint-based data representation with intelligent data-centric augmentation finds a golden mean between performance requirements and practical limitations and forms a strong basis for an accurate and real-time feasible ISL recognition system [4, 11].

2) Data Preprocessing: The skeletal keypoint sequences extracted using MediaPipe sometimes have missing values when landmarks are not detected, which appear as NaNs. To keep sequences consistent, these missing values are replaced with zeros. This preserves the temporal structure while marking points where detection failed.

$$K_{\text{clean}}[i,j] = \begin{cases} 0.0 & \text{if } K_{\text{raw}}[i,j] = \text{NaN} \\ K_{\text{raw}}[i,j] & \text{otherwise} \end{cases} \tag{1}$$

Here, $K_{\text{raw}} \in \mathbb{R}^{30\times144}$ represents a raw keypoint sequence for a single sign. Each of the 30 rows corresponds to a frame, and the 144 columns represent 48 landmarks with 3D coordinates (x, y, z). Since signing speed varies among individuals, all sequences are standardized to 30 frames. Shorter sequences are padded with zeros, while longer ones are truncated. This ensures that every sample has the same temporal length.

To reduce differences in body size and proportions, each feature is normalized using statistics computed across the entire dataset:

$$K_{\text{normalized}} = \frac{K_{\text{clean}} - \mu_{\text{global}}}{\sigma_{\text{global}} + \epsilon} \tag{2}$$

where μ_{global} and σ_{global} are the global mean and standard deviation, and $\epsilon = 10^{-8}$ prevents division errors. After this step, all features are centered around zero with unit variance, which helps the model train more reliably. Each sign label is converted into an integer and then one-hot encoded for multi-class classification. The dataset is split using stratified sampling, with 80% for training and 20% for testing, ensuring balanced class representation. After preprocessing, the dataset is organized as real-valued tensors with dimensions:

$$\mathbb{R}^{N\times T\times F} \tag{3}$$

Here, $N = 4,250$ is the total number of sequences, $T = 30$ represents the number of frames per sequence, and $F = 144$ corresponds to spatial keypoint features. This format maintains the spatiotemporal information while being optimized for deep learning models.

3) Recognition Model: The core of our ISL recognition system is a hybrid deep learning architecture (Fig. 2) that synergistically combines a 1D Convolutional Neural Network (CNN) for local spatio-temporal feature extraction with a Transformer encoder for global temporal dependency modeling. This design is motivated by the distinct strengths of each component in addressing the challenges of sign language recognition. While CNNs are highly effective as spatial feature extractors [5, 6], they are inherently limited in modeling long-range temporal dynamics, which are crucial for distinguishing dynamic signs [9]. Hybrid CNN-LSTM models have been a popular solution to this, achieving notable success [1, 2]. However, LSTM-based models process sequences sequentially, which can be a bottleneck for capturing very long-range contextual relationships and for training efficiency [3, 14]. To overcome these limitations, we replace the recurrent module with a Transformer encoder. The self-attention mechanism of the Transformer allows the model to weigh the importance of all frames in a sequence simultaneously, providing a more powerful and parallelizable method for temporal modeling [3, 12]. This approach has shown superior performance in capturing the temporal evolution of gestures compared to LSTM-based baselines [14]. Our model leverages a custom CNN front-end to distill local motion patterns, the output of which is then processed by the Transformer network to understand the global context of the sign gesture.

a) Spatial Feature Extraction via 1D CNN: The CNN layers capture local motion patterns and reduce the dimensionality:

- **Initial Feature Learning:** A 1D convolutional layer with 64 filters of kernel size 3, followed by a ReLU activation, captures local temporal dependencies between adjacent frames:

$$\mathbf{F}_1 = \text{ReLU}\left(\text{Conv1D}_{64,k=3}(\mathbf{X})\right), \quad \mathbf{X} \in \mathbb{R}^{T\times F} \tag{4}$$

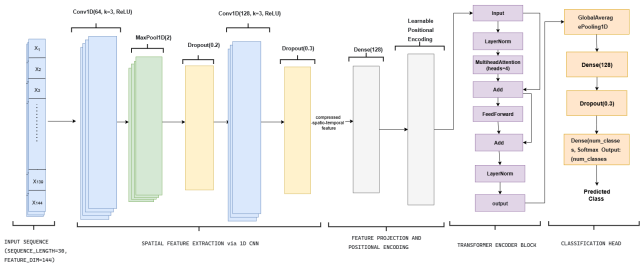

Fig. 2: Overview of the proposed CNN-Transformer hybrid architecture..

where $\mathbf{X}$ is the input sequence of $T = 30$ frames with $F = 144$ features per frame, and $\mathbf{F}_1 \in \mathbb{R}^{T \times 64}$ denotes the resulting feature map.

- **Temporal Reduction:** A max pooling operation halves the temporal resolution from 30 to 15 frames:

$$\mathbf{F}_2 = \text{MaxPool1D}_{p=2}(\mathbf{F}_1), \quad \mathbf{F}_2 \in \mathbb{R}^{T' \times 64}, \quad T' = 15 \tag{5}$$

- **Feature Refinement:** A second convolutional layer with 128 filters extracts more complex features:

$$\begin{aligned} \mathbf{F}_{\text{CNN}} &= \text{ReLU}\left(\text{Conv1D}_{128,k=3}(\mathbf{F}_2)\right) \\ \mathbf{F}_{\text{CNN}} &\in \mathbb{R}^{T' \times D}, \quad D = 128 \end{aligned} \tag{6}$$

Thus, the CNN output $\mathbf{F}_{\text{CNN}}$ provides a compact spatio-temporal representation with reduced temporal length.

b) Temporal Modeling via Transformer Encoder: The features are then prepared for the Transformer Encoder:

- **Feature Projection and Positional Encoding:** The CNN features are projected to a 128-dimensional latent space using a dense layer. Learnable positional encodings ($\mathbf{PE}_{\text{learned}}$) are then added to inject sequence order information:

$$\mathbf{Z} = \text{Dense}_{128}(\mathbf{F}_{\text{CNN}}) + \mathbf{PE}_{\text{learned}}, \quad \mathbf{Z} \in \mathbb{R}^{T' \times D} \tag{7}$$

Here, $\mathbf{Z}$ represents the position-aware feature embeddings used as input to the Transformer.

- **Multi-Head Self-Attention (MHA):** Given queries (Q), keys (K), and values (V), the attention mechanism is computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{D_k}}\right)V \tag{8}$$

where $D_k = 32$ is the key dimensionality. With 4 heads, multiple attention distributions are computed in parallel, allowing the model to attend to different temporal contexts simultaneously.

- **Residual Connections and Pre-Layer Normalization:** To stabilize training and preserve gradient flow, residual connections are applied after both the attention and feed-forward blocks:

$$\mathbf{Z}^{(1)}_{\text{norm}} = \text{LayerNorm}(\mathbf{Z}) \tag{9}$$

$$\mathbf{H}_1 = \mathbf{Z} + \text{MHA}(\mathbf{Z}^{(1)}_{\text{norm}}, \mathbf{Z}^{(1)}_{\text{norm}}, \mathbf{Z}^{(1)}_{\text{norm}}), \quad \mathbf{H}_1 \in \mathbb{R}^{T' \times D} \tag{10}$$

$$\mathbf{H}_{1,\text{norm}} = \text{LayerNorm}(\mathbf{H}_1) \tag{11}$$

$$\mathbf{H}_2 = \mathbf{H}_1 + \text{FFN}(\mathbf{H}_{1,\text{norm}}), \quad \mathbf{H}_2 \in \mathbb{R}^{T' \times D} \tag{12}$$

Here, $\mathbf{H}_1$ and $\mathbf{H}_2$ denote the outputs of the attention and feed-forward blocks, respectively.

- **Feed-Forward Network (FFN):** The FFN consists of two dense layers with expansion and contraction:

$$\begin{aligned} \text{FFN}(x) &= \text{ReLU}(x\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \\ \mathbf{W}_1 &\in \mathbb{R}^{D \times 256}, \quad \mathbf{W}_2 \in \mathbb{R}^{256 \times D}, \quad D = 128 \end{aligned} \tag{13}$$

c) Classification Head: The Transformer output is aggregated and passed to the classifier:

- **Global Average Pooling:** Aggregates features across all $T' = 15$ frames:

$$\mathbf{g} = \frac{1}{T'}\sum_{t=1}^{T'} \mathbf{H}_2^{(t)}, \quad \mathbf{g} \in \mathbb{R}^D \tag{14}$$

where $\mathbf{H}_2^{(t)}$ represents the feature vector at time step t.

- **Dense Classification Layers:** A fully connected layer with ReLU activation and dropout projects $\mathbf{g}$ into the final classification space:

$$\mathbf{h} = \text{ReLU}(\mathbf{g}\mathbf{W}_3 + \mathbf{b}_3), \quad \mathbf{W}_3 \in \mathbb{R}^{D \times D} \tag{15}$$

$$\hat{\mathbf{y}} = \text{softmax}(\mathbf{h}\mathbf{W}_c + \mathbf{b}_c), \quad \mathbf{W}_c \in \mathbb{R}^{D \times C} \tag{16}$$

where $C = 38$ is the number of sign classes, and $\hat{\mathbf{y}}$ is the predicted probability distribution over classes.

4) Design Rationale and Efficiency Considerations:

- **Computational Efficiency:** Reducing the sequence length from 30 to 15 frames lowers the self-attention complexity from $\mathcal{O}(30^2)$ to $\mathcal{O}(15^2)$, achieving a 4× reduction in computational requirements while maintaining essential temporal information.
- **Parameter Optimization:** With approximately 250,000 trainable parameters, the model balances representational capacity with inference efficiency, making it suitable for real-time applications on commodity hardware.
- **Training Stability:** The Pre-LayerNorm configuration with residual connections ensures stable gradient flow during training, while progressive dropout (0.2 → 0.3) prevents overfitting in deeper layers.
- **Temporal Modeling:** The hybrid architecture leverages CNN efficiency for local feature extraction while harnessing the Transformer's global receptive field for comprehensive temporal dependency modeling, addressing the limitations of CNN-LSTM approaches in capturing long-range gesture dynamics.

B. TExt-to-ISl Animation

The Text-to-ISL translation module converts input English text into animated Indian Sign Language through a streamlined, multi-stage pipeline. It starts by processing input text with spaCy to perform part-of-speech tagging and linguistic analysis. The core of the system applies ISL grammatical rules, converting the English SVO structure into ISL's SOV order

while also eliminating out auxiliary verbs and normalizing pronouns. This reordered sequence is then mapped to ISL glosses using a hierarchical strategy that prioritizes multi-word phrases for natural signing of common expressions like greetings, before falling back to single-word mappings and character-level fingerspelling for unknown vocabulary. Finally, the gloss sequence drives the animation engine, which retrieves and concatenates pre-recorded keypoint data to render a smooth, skeletal animation of the signed sentence, delivering a complete ISL video output from the original text.

1) Text Preprocessing and Common Phrase Detection: The input English text first undergoes preprocessing where it is converted to lowercase and tokenized using spaCy's English model. The system then checks for common greetings and phrases (e.g., "good morning", "thank you", "hello") using a predefined dictionary. If detected, these phrases are directly mapped to their corresponding ISL gloss representations without grammatical reordering, while the remaining text proceeds to grammatical processing.

2) ISL Grammatical Processing: For text not matching common phrases, the system applies ISL grammatical rules through the following steps:

1) **POS Tagging and Auxiliary Verb Filtering**: Each token is assigned a part-of-speech tag using spaCy's linguistic analysis, and all auxiliary verbs (is, am, are, was, were, do, does, did, have, has, had, will, shall) are removed from the sequence.
2) **Word Categorization**: Remaining words are categorized into:
   - Subjects: Pronouns and first-occurring nouns (with pronoun normalization using mapping: I/me/my→me, you/your→you, etc.)
   - Objects: Subsequent nouns
   - Verbs: Main verbs (auxiliaries already filtered)
   - Adjectives: Non-color descriptive words
   - Colors: Color terms identified through a predefined color dictionary
   - Negations: Negative markers (not, no, never, nothing)
   - Questions: Question words (what, where, when, why, how, which, who, whom)
   - Others: Adverbs, prepositions, and remaining words
3) **SOV Reordering**: Words are rearranged into ISL's Subject-Object-Verb structure with the following order:

$$\text{Subject} + \text{Object} + \text{Verb} + \text{Adjectives} + \text{Others} + \text{Colors} + \text{Negation} + \text{Questions}$$

3) Gloss Sequence Generation: The reordered token sequence is converted to ISL glosses using a hierarchical lookup strategy:

The algorithm employs a trie data structure for efficient phrase matching, following the workflow shown in Figure **??**. The process begins by initializing counters and proceeds token-by-token through the input sequence. For each position, it first attempts to find the longest matching multi-word phrase in the phrase trie. If a phrase match is found, the corresponding gloss is added to the sequence and the index advances by the phrase length. If no phrase match exists, the algorithm checks for single-word mappings in the gloss map. For unrecognized words, it falls back to character-level finger-spelling, ensuring complete vocabulary coverage through graceful degradation.

4) Animation Generation and Video Output: The final gloss sequence drives the animation generation pipeline, transforming linguistic representations into visual ISL output:

1) **Keypoint Data Retrieval**: Each gloss in the sequence is mapped to its corresponding pre-recorded keypoint animation file through the gloss map dictionary. The keypoint data is extracted from ISL videos using MediaPipe Holistic, which provides 33 pose, 42 hand (21 per hand), and 468 facial landmarks stored in JSON format.
2) **Animation Concatenation**: Pose data for all glosses is concatenated into a continuous sequence maintaining temporal coherence and natural signing rhythm between gestures.
3) **Skeletal Rendering**: For each frame, the system renders:
   - Body pose connections for torso and limbs using MediaPipe landmarks
   - Hand landmarks and connections for precise signing gestures
   - Simplified facial representation
   - Color-coded joints and connections for visual clarity
4) **Video Encoding**: The final animation is encoded using H.264 compression at 25 FPS with 512×512 resolution, producing smooth, continuous ISL video output.

The system ensures linguistic correctness by maintaining ISL's grammatical structure while providing natural signing flow through optimized keypoint sequence concatenation and expressive skeletal animation.

IV. Results and Discussion

This section presents comprehensive experimental results from multiple development phases, demonstrating the efficacy of the DeepISL framework for Sign Language Recognition and Translation (SLRT) research challenges. The proposed framework was rigorously evaluated across different stages of development, with particular emphasis on real-time performance, translation accuracy, and bidirectional capabilities.

A. Evaluation

Furthermore, we plotted the confusion matrix for obtaining the classification performance. The accuracy and loss curves provide additional insights into model training. The confusion matrix results are shown in Figure 3.

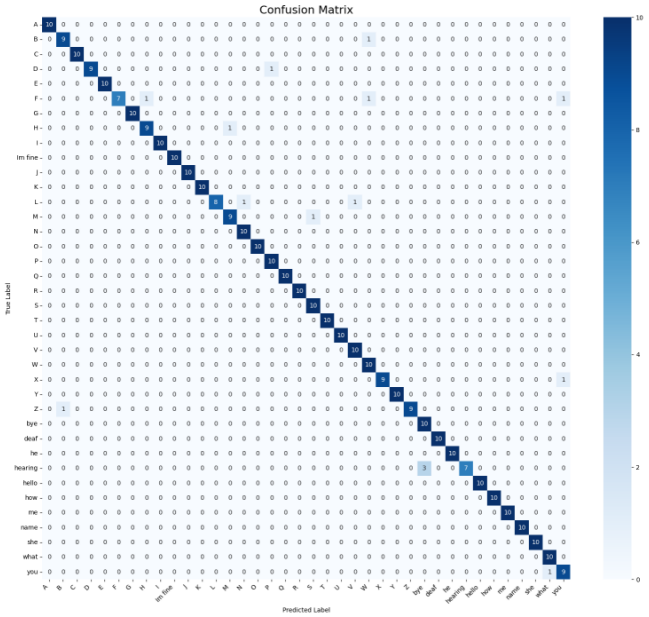

Fig. 3: Confusion Matrix

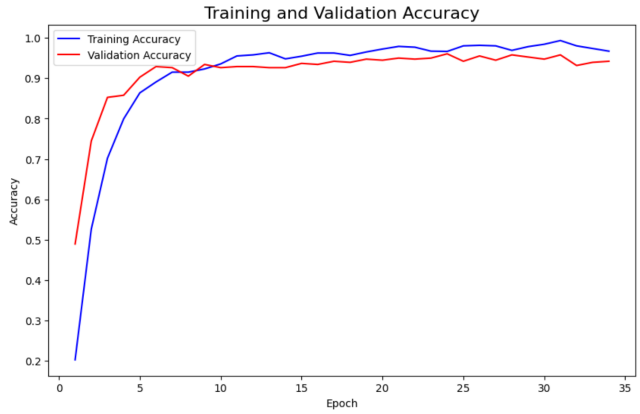

Fig. 4: Accuracy Curves

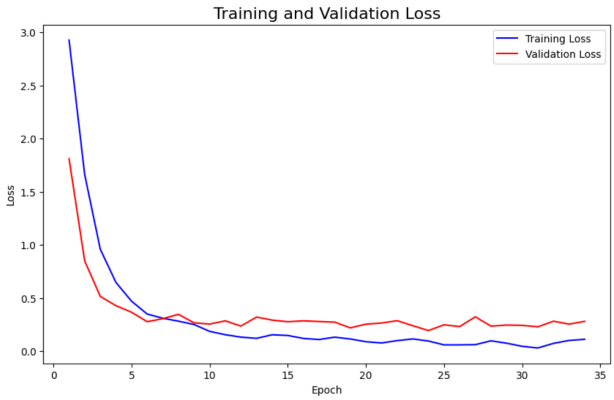

Fig. 5: Loss Curves

B. Comparative Model Superiority

The CNN-Transformer hybrid is superior to pure recurrent approaches. Table I demonstrates how the parallel processing of the Transformer significantly improves F1-score and inference speed, crucial for real-time applications.

TABLE I: Architectural Performance Comparison

Architecture	F1-Score	Latency (ms)	Temporal Method
CNN-LSTM [5]	≈ 0.88	$40 - 60$	Sequential
Pure Transformer	≈ 0.91	$25 - 45$	Attention
Hybrid CNN-Transformer	**0.96**	$\mathbf{20 - 40}$	Attention (Parallel)

C. Real-Time Deployment Performance

The practical utility of DeepISL hinges on its optimized low-latency architecture, which enables seamless real-time interaction through a carefully designed client-server protocol:

1) **Client-Side Processing**: The web application utilizes MediaPipe Holistic (JavaScript) to capture video frames and extract 144D keypoint vectors. Through frame skipping mechanisms (transmitting every 4th frame via `FRAME_SKIP`) and sequence buffering (30 consecutive keypoint vectors), the system maintains optimal performance.
2) **Efficient Communication**: WebSocket streaming via SocketIO enables low-overhead transmission of complete 30×144 sequences to the Flask server through the `predict_sequence` event, significantly reducing latency compared to traditional HTTP requests.
3) **Server-Side Optimization**: The Python server implements global normalization using μ_{global} and σ_{global} statistics, followed by highly optimized TFLite inference for rapid sign classification with confidence scoring.
4) **Intelligent History Management**: The system maintains per-user sign histories with confidence thresholding and duplicate prevention, ensuring accurate sentence construction while preventing server overload through event throttling.

D. Bidirectional Translation Demonstration

The framework's bidirectional capabilities are demonstrated through both recognition (ISL $\rightarrow$ Text) and generation (Text $\rightarrow$ ISL) pipelines, as shown in Figure

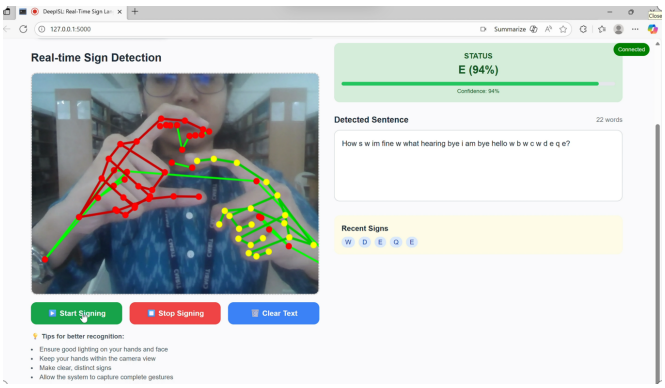

Fig. 6: Real-Time ISL Recognition: MediaPipe keypoint detection with live confidence scoring (94% for sign 'E'), demonstrating robust pose estimation and instant feedback.

Fig. 7: Text-to-ISL Generation: Grammatically correct SOV reordering for "What is your name?" (YOU → NAME → WHAT), showcasing proper ISL syntactic structure.

The system successfully processes input text "What is your name?" by applying ISL grammatical rules to produce the correct gloss sequence YOU → NAME → WHAT, demonstrating effective auxiliary verb elimination and SOV reordering before animation rendering.

V. Conclusion

In this paper, we presented DeepISL, a bidirectional Indian Sign Language (ISL) to English translation framework. The core of our recognition system is a hybrid CNN-Transformer model, which effectively combines the local feature extraction capabilities of convolutional layers with the global temporal modeling of the Transformer's self-attention mechanism. This architecture was validated on a dataset of isolated signs, achieving a high recognition accuracy of 96.05% , demonstrating its strong potential for the task. The integration of a rule-based linguistic module successfully enabled the translation of recognized signs into syntactically correct English phrases. However, the current system has limitations that must be addressed for practical deployment. The vocabulary is constrained to 38 isolated signs, which limits its utility in real-world conversational contexts. Furthermore, the model's performance can be affected by gestural ambiguities and occlusions, and the deterministic nature of the translation module restricts its ability to handle complex semantic contexts. Future work will focus on scaling the system to support a significantly larger vocabulary and transitioning from isolated sign recognition to Continuous Sign Language Recognition (CSLR). To enhance robustness, we plan to integrate multimodal features, including facial expressions to resolve ambiguities and mitigate occlusion. The rule-based translation engine will be replaced by a neural machine translation (NMT) pipeline to manage complex linguistic structures. Finally, the implementation of a text-to-speech (TTS) module will complete the communication loop, providing a fully accessible three-way translation system: ISL to Text to Speech.

References

[1] S. Katoch, V. Singh, and U. S. Tiwary, "Indian Sign Language Recognition System Using SURF with SVM and CNN," *Array*, vol. 14, 2022, doi: 10.1016/j.array.2022.100141.

[2] Q. M. Areeb, Maryam, M. Nadeem, R. Alroobaea, and F. Anwer, "Helping Hearing-Impaired in Emergency Situations: A Deep Learning-Based Approach," *IEEE Access*, 2022, doi: 10.1109/ACCESS.2022.3142918.

[3] B. Natarajan, E. Rajalakshmi, R. Elakkiya, K. Kotecha, A. Abraham, L. A. Gaballa, and V. S. Swamy, "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, doi: 10.1109/ACCESS.2022.3210543.

[4] S. Sharma and S. Singh, "A spatio-temporal framework for dynamic Indian Sign Language recognition," *Wireless Personal Communications*, 2023, doi: 10.1007/s11277-023-10730-8.

[5] G. Khartheesvar, M. Kumar, A. K. Yadav, and D. Yadav, "Automatic Indian Sign Language Recognition Using MediaPipe Holistic and LSTM Network," *Multimedia Tools and Applications*, 2023, doi: 10.1007/s11042-023-17361-y.

[6] S. Hon, M. Sidhu, S. Marathe, and T. A. Rane, "Real-Time Indian Sign Language Recognition Using Convolutional Neural Network," *Multimedia Tools and Applications*, 2024, doi: 10.1007/s11042-024-20384-8.

[7] A. Ghorai, U. Nandi, C. Changdar, T. Si, M. M. Singh, and J. K. Mondal, "Indian Sign Language Recognition System Using Network Deconvolution and Spatial Transformer Network," *Neural Computing & Applications*, 2023, doi: 10.1007/s00521-023-08860-y.

[8] A. Ghorai, U. Nandi, M. M. Singh, C. Changdar, B. Paul, P. Chowdhuri, and P. Pal, "TSI-CNN-Net: Truly Shift-Invariant Convolutional Neural Network for Indian Sign Language Recognition System," *Pattern Analysis and Applications*, 2025, doi: 10.1007/s10044-025-01428-7.

[9] S. Das, S. K. Biswas, and B. Purkayastha, "A deep sign language recognition system for Indian Sign Language," *Neural Computing and Applications*, 2022, doi: 10.1007/s00521-022-07840-y.

[10] S. Das, S. K. Biswas, and B. Purkayastha, "Occlusion Robust Sign Language Recognition System for Indian Sign Language Using CNN and Pose Features," *Multimedia Tools and Applications*, 2024, doi: 10.1007/s11042-024-19068-0.

[11] N. Bansal and A. Jain, "Word Recognition from Indian Sign Language in Videos Using Dual Feature Descriptor and GMT-MASKRCNN Recognition Technique," *Multimedia Tools and Applications*, 2024, doi: 10.1007/s11042-024-20384-8.

[12] R. Sreemathy, P. Chordiya, S. Khurana, et al., "Sign Language Video Generation from Text Using Generative Adversarial Networks," *Optical Memory and Neural Networks*, vol. 33, pp. 466–476, 2024, doi: 10.3103/S1060992X24700851S.

[13] R. Sreemathy, P. Chordiya, S. Khurana, and M. Turuk, "Sign Language Video Generation from Text Using Generative Adversarial Networks," *Optical Memory and Neural Networks*, 2024, doi: 10.3103/S1060992X24700851.

[14] F. M. Najib, "Sign Language Interpretation Using Machine Learning and Artificial Intelligence," *Neural Computing & Applications*, 2024, doi: 10.1007/s00521-024-10395-9.

[15] S. Sabharwal and P. Singla, "Translation of Indian Sign Language to Text - A Comprehensive Review," *Neural Computing & Applications*, 2024, doi: 10.1007/s00521-024-10395-9.