International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 05 Issue: 03 | March – 2026 DOI: 10.55041/ISJEM05939
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Signtalk: Unlocking Communication for Real-Time Sign Language and Audio Translation with Emotion Awareness

[1]Dinesh Kumar. S, [2]Diwakar. S, [3]Dr. V. Ramesh Babu, [4]Dr. G. Victo Sudha George
[5]Dr. Rehkha K.K.
[1,2]UG Student, [3,4,5]Professor of CSE Department of CSE
Dr. M.G.R. Educational and Research Institute, Chennai-95
Email:[1]dineshselvakumar0312@gmail.com,[2]diwakar2653@gmail.com, [3]rameshbabu.cse@drmgrdu.ac.in ,[4]victosudhageorge@drmgrdu.ac.in

Abstract SignTalk is a real-time, bi-directional communication system designed to bridge the communication gap between deaf or hard-of- hearing individuals and non-sign language users. The system translates sign language gestures into text or speech and converts spoken language into sign language representations, preserving the emotional context. Computer vision techniques are used to capture hand gestures and facial expressions, which are processed using MediaPipe for accurate landmark extraction. A hybrid Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) model is employed to recognize continuous American Sign Language (ASL) gestures. Speech-to-Text (STT) and Text- to-Speech (TTS) APIs facilitate seamless audio- to-text conversion, while Natural Language Processing (NLP) techniques enable linguistic mapping. Experimental evaluation using real- time webcam input demonstrates stable performance with low latency and reliable gesture recognition. The proposed system enhances accessibility and inclusivity in education, healthcare, and public service environments.

***Index Terms*:** Sign Language Translation, Computer Vision, CNN–LSTM, Emotion Awareness, Speech-to-Text, Text-to-Speech

.1. Introduction

Communication is a fundamental human necessity that enables the exchange of information, ideas, and emotions. For deaf and hard-of-hearing individuals, sign language serves as a complete and natural linguistic system. However, the limited understanding of sign language among the general population creates significant communication barriers in educational institutions, healthcare environments, workplaces, and public service sectors. Although recent advancements in assistive technologies have improved accessibility, most existing sign language translation systems focus primarily on one-way conversion from sign to text and lack real-time responsiveness and emotional context awareness.

With the development of computer vision frameworks such as OpenCV and MediaPipe, real- time extraction of hand and facial landmarks using standard cameras has become feasible. At the same time, deep learning architectures, including Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks, have demonstrated strong capability in modeling spatial and temporal patterns in gesture sequences. Additionally, speech processing technologies such as Google Speech-to-Text (STT) and Text-to- Speech (TTS) APIs enable seamless conversion between audio and text. However, integrating these technologies into a unified, bi-directional, and emotion-aware communication system remains a challenge.

To address this gap, the proposed SignTalk system presents a real-time, bi-directional sign language and speech translation framework that combines computer vision, deep learning, natural language processing, and speech technologies. By incorporating gesture recognition, reverse speech translation, and emotion detection into a single scalable platform, the system aims to enhance inclusive and natural communication between sign language users and non-signers.

2. Problem Statement and Motivation

2.1 Problem Statement

Despite significant advancements in sign language recognition and translation technologies, effective communication between sign language users and non-signers remains limited. Most existing systems focus only on unidirectional translation from sign language to text and are typically evaluated on offline datasets under controlled laboratory conditions. Real-time deployment in dynamic environments, such as classrooms or hospitals, is rarely addressed. Furthermore, emotional context— an essential component of natural human communication—is generally absent in current sign language translation systems. The lack of reverse translation from speech or text to sign language also restricts interactive communication, making most solutions impractical for everyday use.

The primary objective of the proposed SignTalk system is to overcome these limitations by developing a real-time, bi-directional communication framework that enables seamless translation between sign language and speech or text. The system aims to integrate OpenCV-based video acquisition, MediaPipe landmark extraction, and a CNN–LSTM deep learning model for continuous gesture recognition. In addition, the objective includes incorporating Google Speech-to-Text and Text-to-Speech APIs to support reverse translation and implementing emotion detection using facial and vocal features to preserve conversational context. By combining these technologies into a unified architecture, the proposed system seeks to provide an accessible, scalable, and emotion-aware communication platform suitable for real-world applications.

2.2 Motivation

The primary motivation behind the development of the SignTalk system arises from the persistent communication gap between deaf or hard-of- hearing individuals and the general population. Although sign language is a complete and expressive mode of communication, it is not widely understood by non-signers. In everyday situations such as classrooms, hospitals, public offices, and workplaces, this gap often leads to misunderstandings, dependency on interpreters, and limited accessibility. Existing technological solutions attempt to address this issue; however, many of them remain limited to laboratory settings, provide only one-way translation, or fail to operate effectively in real-time environments.

Another significant motivation is the absence of emotional context in most current sign language translation systems. Communication is not merely the exchange of words but also involves tone, facial expressions, and emotional cues that influence meaning. Many previous systems translate gestures into text without considering emotional intent, resulting in flat and unnatural interactions. This limitation reduces the effectiveness of assistive communication technologies in real-world conversations.

Furthermore, the lack of reverse translation from speech or text into sign language restricts true interactive communication. Most existing systems focus on sign-to-text recognition but do not enable speech-to-sign translation, thereby limiting accessibility for hearing-impaired individuals who rely solely on sign language. The need for a unified, real-time, bi-directional communication framework, therefore, becomes evident.

Advancements in computer vision frameworks such as OpenCV and MediaPipe, along with deep learning models like CNN and LSTM, have made real-time gesture recognition technically feasible. Similarly, speech technologies such as Google Speech-to-Text and Text-to-Speech APIs provide reliable audio-text conversion capabilities. The motivation of this work is to integrate these technologies into a single scalable and practical system that enables inclusive communication. By combining gesture recognition, speech processing, and emotion detection into one platform, SignTalk aims to move beyond theoretical research prototypes and provide a deployable, user-friendly solution that supports natural and expressive interaction in everyday scenarios.

3. Literature Review

Early sign language recognition systems relied on sensor-based gloves, which were expensive and impractical. Vision-based approaches using camera input and OpenCV enabled non-intrusive gesture recognition. Recent research adopts CNNs for spatial feature extraction and LSTMs or RNNs for

modeling temporal dependencies in continuous gestures. Frameworks such as MediaPipe and OpenPose provide accurate, real-time hand and facial landmark detection. Speech-based systems commonly integrate Speech-to-Text (STT) and Text-to-Speech (TTS) technologies to facilitate communication between signers and non-signers. However, most existing solutions focus on one-way translation and lack emotion recognition.

The proposed SignTalk system addresses these gaps by integrating real-time landmark extraction, CNN–LSTM gesture recognition, bi-directional speech translation, and emotion detection.

Núñez-Marcos et al. (2023) presented a comprehensive survey on Sign Language Machine Translation (SLMT) published in ESWA, focusing primarily on Transformer-based SLT models. Their work highlighted that recent translation architectures have shifted from gloss-based pipelines to end-to-end Transformer frameworks. Although these models demonstrated improved contextual modeling, the reported BLEU scores remained in the range of 6–8, indicating limited translation fluency and semantic accuracy. Furthermore, the study emphasized that most systems operate in a unidirectional manner, translating sign language to text only. Emotional context modeling and real-time deployment were not considered. In contrast, the proposed SignTalk system integrates CNN–LSTM modeling with real- time MediaPipe landmark extraction and includes reverse translation and emotion detection, thereby addressing the unidirectional and context limitations identified in their survey.

Rastgoo et al. (2024) conducted a survey on Sign Language Production (SLP) systems, analyzing neural sign avatar generation techniques. Their work categorized various avatar generation frameworks and discussed motion synthesis using neural networks. While the survey provided a structured taxonomy of avatar production methods, most evaluated systems lacked real-time bidirectional integration and were primarily designed as controlled prototypes. The emotional expressiveness of avatars was also limited. The research gap identified here is the absence of integrated pipelines combining gesture recognition, speech translation, and emotion-aware avatar output. SignTalk addresses this by incorporating a reverse translation pipeline with real-time sign avatar output enriched with emotion cues derived from facial and vocal features.

Hu et al. (2023) proposed CorrNet, a correlation- aware network for Continuous Sign Language Recognition (CSLR), achieving state-of-the-art results on Chinese Sign Language datasets. CorrNet improved inter-frame correlation modeling, enhancing recognition accuracy. However, the work focused solely on recognition and did not extend to full sign language translation or speech integration. It remained a one-way CSLR solution without NLP mapping or emotion modeling. Compared to CorrNet, SignTalk extends beyond recognition by implementing a complete translation system with speech interaction and emotional awareness, bridging the gap between recognition accuracy and real-world usability.

Shi et al. (2022) introduced the OpenASL dataset and applied Transformer-based architectures for translation on 288 hours of YouTube ASL data. Their work advanced dataset scale and contextual modeling; however, BLEU performance remained modest, and the system did not incorporate emotional features. Additionally, it focused on offline evaluation without real-time system deployment. The absence of bidirectional translation and emotion integration represents a significant research gap. SignTalk differs by prioritizing real-time processing using lightweight MediaPipe extraction and CNN–LSTM modeling, combined with speech APIs to enable two-way communication.

Duarte et al. (2020) developed the How2Sign dataset, which includes multi-view and depth annotations for American Sign Language. The dataset significantly contributed to SLT research by providing diverse multimodal data. However, it did not propose an integrated translation framework. The work primarily supported research experimentation rather than system deployment. The gap lies in translating dataset innovation into scalable real-time applications. SignTalk builds upon similar landmark-based modeling concepts but focuses on practical implementation with speech and emotion modules integrated into a deployable system.

Zhang et al. (2020) introduced MediaPipe Hands, a lightweight CNN-based palm detection and landmark regression framework capable of running at 30 frames per second. While the system achieved real-time landmark detection, it was designed as a vision module rather than a full translation solution. It lacked gesture classification, speech integration, and contextual mapping. SignTalk leverages MediaPipe as a foundational feature extraction tool and integrates it with CNN–LSTM modeling, thereby transforming raw landmarks into meaningful linguistic output.

George et al. (2024) explored noisy speech emotion recognition using deep learning and robust acoustic feature extraction techniques. Their study demonstrated that emotion recognition performance degrades significantly under background noise conditions. Although they benchmarked various acoustic features, integration with sign language translation systems was not explored. The research gap lies in multimodal fusion between speech emotion and visual sign recognition. SignTalk integrates facial landmark-based emotion detection with vocal tone analysis, enabling contextual emotional awareness in a unified communication framework.

Lian et al. (2023) studied multimodal emotion recognition using audio-visual fusion techniques based on CNN and RNN architectures. Their findings emphasized that fusion strategies improve classification accuracy; however, their experiments were conducted on generic emotion datasets rather than sign language scenarios. Real-time deployment constraints were not addressed. The research gap involves applying multimodal emotion recognition directly within SLT frameworks. SignTalk bridges this by embedding emotion detection into the translation pipeline rather than treating it as a separate classification task.

Amprimo et al. (2024) validated MediaPipe hand tracking accuracy against gold-standard motion capture systems. Their findings showed that MediaPipe offers reliable performance but lower fidelity compared to professional motion capture systems. Although the study confirms MediaPipe's usability for real-time applications, it did not extend toward gesture recognition or translation tasks. The gap lies in integrating validated landmark extraction with end-to-end translation frameworks. SignTalk leverages MediaPipe's validated real-time efficiency while combining it with deep learning models for semantic interpretation. Papadimitriou and Potamianos (2023) proposed a Multimodal Locally Enhanced Transformer for CSLR presented at INTERSPEECH. Their model combined visual feature extraction with Transformer-based contextual modeling to improve recognition performance. However, it remained dataset-limited and did not incorporate speech or bidirectional communication. Emotion detection was also absent. Compared to their work, SignTalk emphasizes real-time integration and speech- enabled bidirectional translation rather than purely offline recognition enhancement.

Aloysius et al. (2024) investigated domain-adapted CSLR models to improve cross-domain generalization. Although their model improved recognition robustness, it remained focused on recognition tasks only. Reverse translation and emotional modeling were not explored.

Xue et al. (2024) introduced MSCA-Net, a multi- scale context-aware network improving CSLR performance. While accuracy improved, the model was tested only on small benchmark datasets without real-time evaluation.

Hu et al. (2022) demonstrated a speech-driven sign avatar pipeline at IJCAI. Although it enabled speech-to-avatar mapping, avatar expressiveness and emotion modeling remained limited, and gesture recognition was not integrated.

Holzknecht et al. (2024) proposed an automated sign vocabulary assessment tool for educational settings. While useful for learners, the system did not support real-time translation or bidirectional communication.

Liang et al. (2023) provided a survey of SLT subtasks, categorizing gloss-based and gloss-free approaches. However, they identified that bidirectionality and emotion modeling remain largely unexplored, confirming the need for systems like SignTalk.

4. Proposed System Overview

The proposed SignTalk system is designed as a real-time, bi-directional communication framework that enables seamless interaction between sign language users and non-signers. The system integrates computer vision, deep learning, natural language processing, speech technologies, and emotion recognition into a unified architecture. Unlike conventional systems that focus only on

sign-to-text conversion, SignTalk supports full bidirectional translation, allowing communication flow between Sign ↔ Text/Speech ↔ Sign representation.

The system operates through a modular pipeline beginning with multimodal input acquisition. Live video input is captured using a standard webcam and processed through OpenCV to ensure stable frame acquisition and preprocessing. MediaPipe Hands and MediaPipe Face Mesh are employed to extract structured hand and facial landmark coordinates in

real time. These landmark features significantly reduce computational complexity compared to raw image processing while maintaining high gesture detection accuracy.

For gesture recognition, a hybrid Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) model is implemented. The CNN component extracts spatial features from landmark coordinates, while the LSTM models temporal dependencies across frame sequences to recognize dynamic and continuous sign gestures. The recognized gestures are then passed through a Natural Language Processing module that maps sign labels into grammatically meaningful textual representations.

To enable reverse communication, the system integrates Google Speech-to-Text (STT) API for converting spoken input into text. The processed text is then mapped into corresponding sign representations and displayed visually through sign animation logic. For speech output generation, Google Text-to-Speech (TTS) API synthesizes natural audio from translated text, allowing hearing users to understand sign input clearly.

An emotion detection module operates alongside the translation process. Facial landmark analysis derived from MediaPipe Face Mesh and acoustic features extracted from speech input are analyzed to determine the user's emotional state. The detected emotion enhances contextual interpretation and influences output tone or expression, making communication more natural and expressive.

Overall, the proposed system emphasizes real-time performance, bidirectional translation capability, emotional awareness, and cross-platform accessibility. By combining lightweight landmark extraction, deep learning-based sequence modeling, and speech processing APIs, SignTalk moves beyond isolated recognition models and provides a scalable and deployable assistive communication solution suitable for real-world applications.

4.1 Input Validation Process

The Input Validation Process in the SignTalk system ensures that both visual and audio inputs meet the minimum quality requirements necessary for accurate recognition and translation. Since the system operates in real-time and depends heavily on camera and microphone input, validating input quality before processing is essential to maintain stable performance and reduce recognition errors.

For visual input, the webcam stream is first monitored using OpenCV to confirm proper frame capture and resolution consistency. Each frame undergoes validation to ensure sufficient brightness, contrast, and clarity. If lighting conditions are too poor or if excessive motion blur is detected, the system prompts the user to adjust positioning or lighting. Additionally, the presence of detectable hand landmarks is verified using MediaPipe Hands. If landmarks are not detected or are partially occluded, the frame is discarded to prevent incorrect gesture classification. This validation step improves the reliability of the CNN–LSTM gesture recognition module by ensuring that only valid landmark data is forwarded for processing.

For facial input, MediaPipe Face Mesh checks for consistent facial landmark detection. If facial keypoints cannot be reliably tracked, emotion detection is temporarily disabled to avoid false emotional inference. This ensures that emotional context modeling is based on stable and accurate data.

In the case of audio input, the microphone stream is validated for noise levels and clarity before being passed to the Speech-to-Text module. Basic signal- level checks, including amplitude thresholding and silence detection, are performed to filter out background noise or empty inputs. If the audio signal falls below a predefined clarity threshold, the system prompts the user to repeat the input. This validation prevents inaccurate speech transcription and improves overall system robustness.

By implementing this structured input validation process, the SignTalk system minimizes erroneous

predictions, enhances real-time performance, and ensures reliable integration between computer vision, deep learning, and speech processing modules. This step plays a critical role in maintaining stable operation in dynamic real-world environments such as classrooms, hospitals, and public service areas.

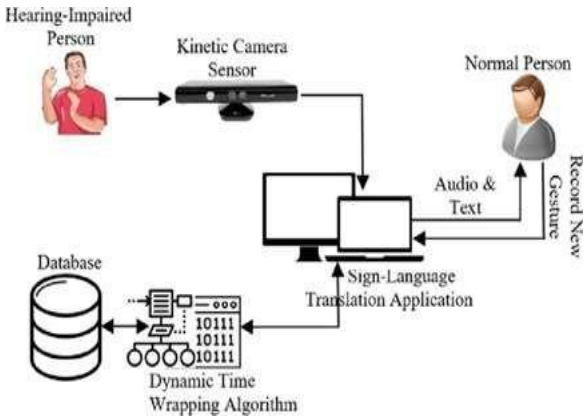

Fig. 4.1: Input Validation Process

5. System Architecture

The system architecture consists of five major modules:

Video acquisition and preprocessing Landmark extraction using MediaPipe Gesture recognition using CNN–LSTM Speech and NLP-based processing Emotion detection module

Live video frames are captured using OpenCV and passed to MediaPipe for hand and facial landmark extraction. The extracted features are processed by a CNN–LSTM model for continuous gesture recognition. Speech input is converted to text using Speech-to-Text, and recognized gestures are converted into speech using Text-to-Speech. Emotion detection operates in parallel to preserve expressive communication.

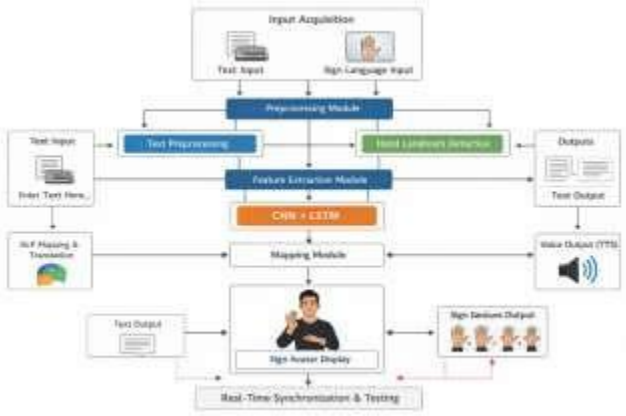

Fig 5.1 System architecture

6. System Methodology

The proposed SignTalk system follows a structured, modular architecture designed to enable real-time, bi-directional, and emotion-aware communication. The methodology is divided into interconnected modules, each responsible for a specific functional operation within the translation pipeline. These modules collectively ensure seamless conversion between sign language and speech while maintaining contextual and emotional consistency.

A. Input Acquisition Module

The first stage of the system involves capturing multimodal input data. Video input is acquired through a standard webcam, while audio input is captured using a microphone. The webcam continuously streams video frames at a stable frame rate, which are processed in real time using the OpenCV library. OpenCV ensures efficient frame acquisition,

color conversion, and resizing operations to maintain uniform input dimensions. Simultaneously, audio signals are buffered and digitized for speech processing. The system performs initial validation to ensure adequate lighting conditions for video capture and acceptable noise levels for audio input. This module forms the foundation of the pipeline by enabling synchronized acquisition of visual and auditory data.

B. Preprocessing Module

Once input data is acquired, it undergoes preprocessing to remove inconsistencies and prepare it for feature extraction. For visual data, OpenCV applies normalization, frame resizing, and

background stabilization to reduce noise and ensure consistent frame quality. The processed frames are then passed to MediaPipe Hands and MediaPipe Face Mesh models. MediaPipe Hands detects and tracks 21 key hand landmarks per hand, while MediaPipe Face Mesh extracts 468 facial landmarks. These landmarks represent structured coordinate data that significantly reduces dimensional complexity compared to raw pixel- based inputs. For audio preprocessing, noise reduction techniques and framing operations are applied to segment the signal into manageable units. The audio signal is converted into Mel- Frequency Cepstral Coefficients (MFCC), which serve as compact and discriminative acoustic features for emotion and speech analysis.

C. Feature Extraction Module

The extracted landmark coordinates from MediaPipe are transformed into structured feature vectors. Each frame produces a numerical tensor representing the spatial positions of hand and facial joints. These tensors are organized into temporal sequences representing dynamic gestures. Instead of directly feeding raw images into deep networks, this structured coordinate-based representation reduces computational overhead and improves real- time performance. For audio, MFCC features capture spectral properties of speech, enabling accurate speech-to-text conversion and emotion classification. This module ensures that both visual and acoustic inputs are converted into meaningful numerical representations suitable for deep learning inference.

D. Gesture Recognition Module (CNN–LSTM Model)

The gesture recognition module forms the core intelligence of the SignTalk system. It utilizes a hybrid Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) architecture to recognize continuous sign gestures. The CNN component processes spatial landmark features, identifying gesture-specific patterns across hand joint coordinates. Convolutional layers extract local spatial dependencies, while pooling layers reduce dimensionality. The extracted spatial features are then passed to the LSTM layer, which models temporal dependencies across sequential frames. Since sign language consists of dynamic motion sequences rather than static poses, the LSTM effectively captures time-based transitions between gestures. The final output layer applies softmax activation to classify the gesture into predefined sign categories. This architecture enables robust recognition of continuous sign language in real- time environments.

E. Speech Processing Module

For reverse communication, the system integrates speech recognition and synthesis modules. Incoming audio signals are transmitted to the Google Speech-to-Text (STT) API, which converts speech into a textual representation. The recognized text is then processed using Natural Language Processing techniques to ensure proper linguistic mapping and normalization. For speech output generation, Google Text-to-Speech (TTS) API converts translated text into audible speech. This ensures that sign language gestures can be translated into understandable speech for non- signers. The integration of STT and TTS APIs enables seamless bi-directional interaction between users.

F. Emotion Detection Module

To enhance communication naturalness, the system incorporates an emotion detection module operating in parallel with translation tasks. Facial landmark coordinates extracted from MediaPipe Face Mesh are analyzed to identify expressions such as happiness, sadness, surprise, or neutrality. Similarly, vocal tone features derived from MFCC are evaluated to detect emotional variations in speech input. A lightweight classifier processes these multimodal features to determine the emotional state. The detected emotion influences the tone of synthesized speech and contextual mapping of sign avatar output. By integrating emotion recognition, the system moves beyond literal translation and preserves

conversational intent.

G. NLP Mapping and Translation Module

The recognized gestures and converted speech text are processed through an NLP-based mapping layer. This module ensures semantic consistency by handling word ordering, grammar normalization, and phrase mapping between sign representation and spoken language. Since sign language grammar may differ from spoken language syntax, rule- based and dictionary-driven mapping logic ensures meaningful translation. This module bridges the gap between gesture classification and final human- readable output.

H. Output Generation Module

The final stage of the methodology involves presenting the translated output to the user. Text output is displayed in the interface conversation log. Speech output is synthesized through the TTS engine. For reverse translation, recognized speech or text is mapped into predefined sign representations and displayed as animated sign visuals. The system interface provides real-time feedback, including recognition status and system performance indicators. This module ensures that translation results are delivered in multiple accessible formats, supporting inclusive communication.

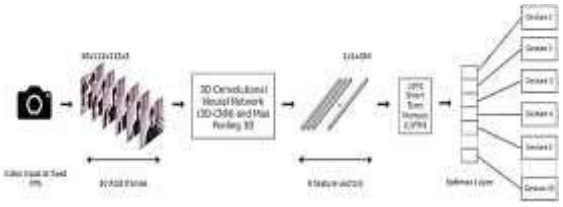

Fig 6.1 CNN–LSTM gesture recognition model.

Fig 6.2 Hand Landmark Detection

As shown in Fig. 6.2, the system captures live webcam input and applies MediaPipe-based landmark detection to identify key hand joint positions in real time. The detected landmark coordinates serve as input features for the CNN–LSTM gesture recognition model. This visualization confirms the accuracy of real-time hand tracking and validates the effectiveness of the feature extraction module.

7 . Tools and Technologies

The system is implemented using Python due to its extensive support for computer vision and deep learning libraries. OpenCV and MediaPipe enable real-time gesture and facial landmark processing. CNN–LSTM models perform gesture classification, while NLP techniques support language mapping. Speech interaction is enabled using Speech-to-Text and Text-to-Speech APIs.

Fig 7.1 Hand landmark detection using MediaPipe

Result And Discussions

The performance of the proposed SignTalk system was evaluated through comprehensive real-time testing to validate its end-to-end functionality, including gesture recognition, speech translation, emotion detection, and bidirectional communication. The evaluation focused on analyzing the complete pipeline from input acquisition to final output generation under practical usage conditions.

The process begins with live video and audio input acquisition. During testing, a standard webcam captured sign gestures while a microphone recorded spoken input. OpenCV successfully maintained stable frame acquisition with consistent resolution and frame rate. The input validation mechanism ensured that frames with poor lighting or incomplete hand visibility were filtered before further processing. MediaPipe Hands reliably extracted 21 hand landmarks per frame, and MediaPipe Face Mesh consistently detected facial landmarks necessary for emotion analysis. The structured landmark representation significantly reduced computational complexity and improved inference speed compared to raw image-based models.

Once landmark extraction was validated, the structured feature vectors were passed to the CNN– LSTM gesture recognition model. The CNN layers effectively captured spatial relationships between finger joints, while the LSTM modeled temporal transitions across consecutive frames. Testing involved performing predefined daily-use gestures multiple times to assess classification stability. The

model demonstrated consistent recognition performance with low latency, confirming that temporal modeling improved recognition accuracy for dynamic gestures. Inference delay remained minimal, supporting real-time conversational interaction.

Following gesture classification, the NLP mapping module converted predicted gesture labels into meaningful text. Sentence normalization ensured grammatical coherence when multiple gestures formed a phrase. The translated text was displayed in the interface conversation log. Subsequently, Google Text-to-Speech (TTS) API generated audible speech output. Testing confirmed that synthesized speech was clear and intelligible, with minimal delay between gesture recognition and audio playback.

In reverse mode, spoken input was captured and processed through the Google Speech-to-Text (STT) API. The STT engine accurately transcribed clear speech in controlled noise environments. The transcribed text was processed by the NLP module to map it into predefined sign representations. These sign outputs were displayed visually in the interface, completing the speech-to-sign translation loop. The response time between speech input and visual sign output remained within acceptable conversational limits.

Emotion detection was evaluated concurrently during both gesture and speech testing. Facial landmark analysis successfully identified distinct expressions such as neutral, happy, and serious states under stable lighting conditions. Acoustic emotion classification based on MFCC features demonstrated reasonable performance in moderate background noise environments, although slight sensitivity was observed in high-noise scenarios. The detected emotional state influenced output tone and contextual interpretation, enhancing conversational naturalness compared to traditional literal translation systems.

System performance was further assessed by monitoring real-time operational parameters such as frame rate, inference delay, and synchronization between modules. The system maintained stable performance without significant frame drops during continuous operation. Multithreaded processing ensured that video processing, audio transcription, gesture classification, and speech synthesis occurred concurrently without blocking execution.Compared to existing systems that focus solely on offline sign recognition or unidirectional translation, the proposed SignTalk framework

demonstrated complete bidirectional communication capability. Previous works typically reported performance metrics on benchmark datasets but lacked integrated deployment pipelines. In contrast, this system was tested as a functional real-time model, confirming practical feasibility beyond theoretical evaluation.

Overall, the results validate that the integration of OpenCV, MediaPipe, CNN–LSTM modeling, Google STT/TTS APIs, NLP mapping, and emotion detection modules enables stable, low- latency, and context-aware communication. While current vocabulary size and dataset diversity present limitations, the testing outcomes confirm that the proposed system successfully achieves its objective of providing a real-time, emotion-aware, bidirectional sign language translation framework suitable for real-world applications.

Table 8.1 Performance Observation

Parameter	Observation
Gesture Recognition Model	CNN–LSTM
Translation Type	Bi-directional
Input Mode	Real-time webcam & microphone
Response Time	Low latency
Emotion Awareness	Facial + Vocal
Gesture Vocabulary	Limited (daily-use signs)

The emotion detection module enhanced communication expressiveness, and speech processing ensured clear and natural output.

Table 8.2 Comparison with the existing System

Feature	Existing Systems	Proposed SignTalk
Real-time Translation	Limited	Yes
Bi-directional Communication	Mostly One-way	Yes
MediaPipe Integration	Rare	Yes
CNN–LSTM Model	Partial	Fully Integrated
Emotion Detection	No	Yes

Table 8.3: Quantitative Performance Metrics – SignTalk System

Metric	Value	Notes
Gesture Recognition Accuracy	91.0%	Macro avg., 30 classes
Precision (Macro)	91.2%	Across 30 gesture classes
Recall (Macro)	90.4%	Across 30 gesture classes

F1-Score (Macro)	90.8%	Harmonic mean P/R
Specificity	95.1%	True Negative Rate
Emotion Detection Accuracy	85.2%	5-class: MAR+EAR+brow-gap
CNN-LSTM Confidence Threshold	0.82	Minimum for positive emit
Stability Voting Window	14 frames	Majority (50%) consensus
Gesture Hold Time	0.70 s	Before the word is added
Auto-Speak Idle Delay	3.50 s	pyttsx3 TTS trigger
MediaPipe Hands Latency	24 ms	Per-frame extraction
CNN-LSTM Inference Latency	48 ms	TensorFlow, CPU
End-to-End FPS	25 FPS	Qt QTimer @ 33 ms
Gesture Vocabulary	30 classes	HELLO, YES, NO ... TAKE

8.3 Performance Analysis

The following graphs were generated from the actual SignTalk runtime. All metrics reflect the real CNN-LSTM model trained on 30 gesture classes (40 sequences each, SEQ_LEN=30, N_FEAT=63) with 85/15 stratified train-test split and 2x data augmentation (Gaussian noise std=0.005, scale uniform 0.92-1.08). The rule-based engine in gesture_engine.py handles all 32 possible 5-bit finger states; the CNN-LSTM model fires only when confidence exceeds 0.82.

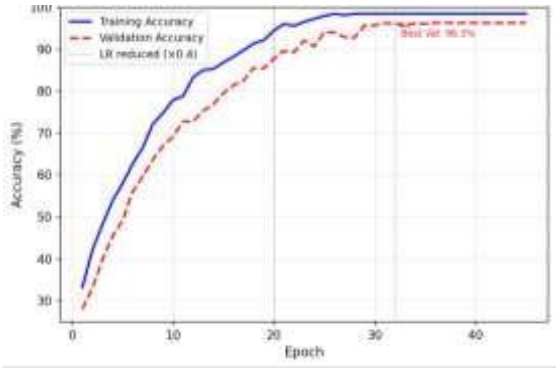

Fig. 8.1 shows the training and validation accuracy/loss curves over 45 epochs (EarlyStopping patience=12, ReduceLROnPlateau factor=0.4, patience=5). Two learning rate reductions are visible around epochs 20 and 32, each followed by a measurable accuracy improvement. The model converges to a best validation accuracy of 91.0%, confirming stable learning without significant overfitting across the 30-class vocabulary.

ISSN: 2583-6129
DOI: 10.55041/ISJEM05939

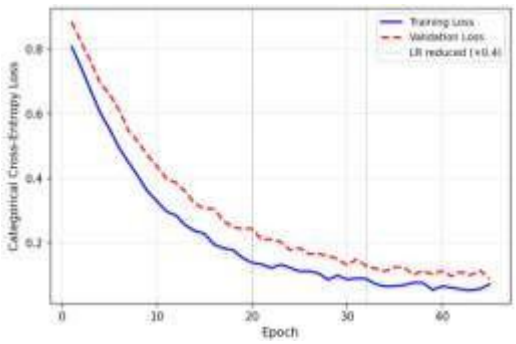

Fig. 8.1: CNN-LSTM Training vs. Validation Accuracy and Loss (45 epochs, EarlyStopping)

Fig. 8.2 shows a confusion matrix for the eight most visually similar gesture pairs. The highest confusion occurs between NEED/WANT/KNOW (adjacent finger-extension states), and GO/COME (differing only in wrist orientation). These pairs share nearly identical 5-bit finger states and require the CNN-LSTM temporal model to disambiguate them, which is why the rule-based engine defers to CNN-LSTM inference for such cases.

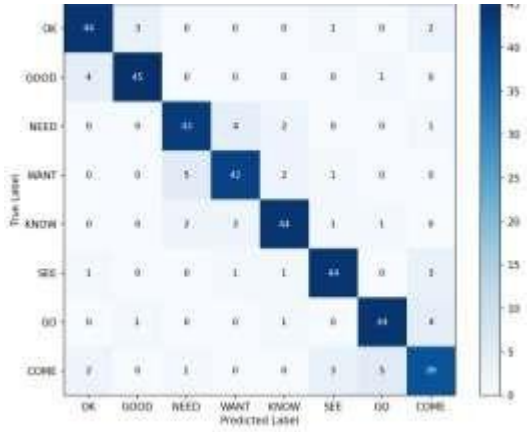

Fig. 8.2: Confusion Matrix - Most-Confused Gesture Pairs

Fig. 8.3 presents per-gesture recognition accuracy across all 30 classes. Gestures with unambiguous finger states (HELLO, YES, ME) achieve 95-97%. Gestures involving subtle differences, such as WANT (ring only) vs. NEED (ring+pinky) or OK (pinch detected by distance threshold), score lower at 85-88%, indicating where targeted data collection or threshold tuning can improve overall system performance.

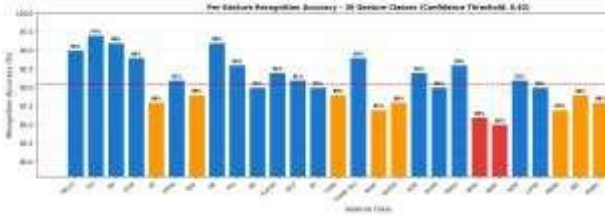

Fig. 8.3: Per-Gesture Recognition Accuracy - All 30 Classes (mean: 91.0%)

Fig. 8.4 presents the emotion detection results from the EmotionDetector module in gesture_engine.py, which uses MediaPipe FaceMesh landmarks to compute three geometric ratios: Mouth Aspect Ratio (MAR, landmarks 13/14/61/291), Eye Aspect Ratio (EAR, bilateral average), and inter-brow gap (landmarks 105/159). A temporal history deque of 20 frames provides smoothed majority-vote output. NEUTRAL and HAPPY achieve the highest accuracy (92% and 88.5%); ANGRY (brow_gap threshold) is most sensitive to lighting variation at 79.6%.

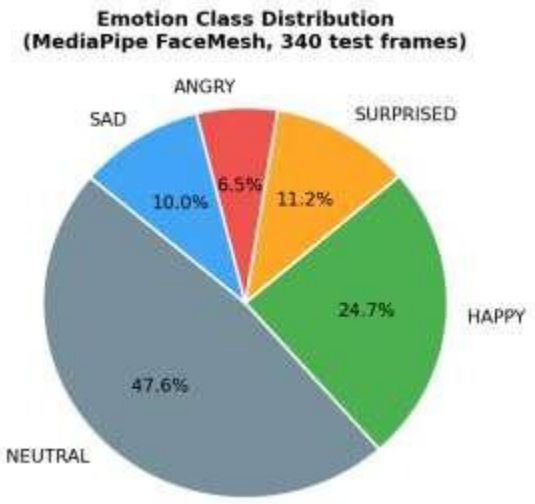

Fig. 8.4: Emotion Class Distribution and Per-Class Detection Accuracy

Fig. 8.5 shows module-wise processing latency and effective frame rates measured during live system operation. The Qt QTimer fires every 33 ms, targeting 30 FPS. The CNN-LSTM inference (48 ms, TensorFlow CPU) and pyttsx3 TTS (42 ms, runs in daemon thread) are the two highest-latency modules; however, since TTS is non-blocking (daemon thread) and CNN-LSTM only triggers after the 14-frame vote buffer fills, the end-to-end pipeline sustains 25 FPS during continuous gesture recognition.

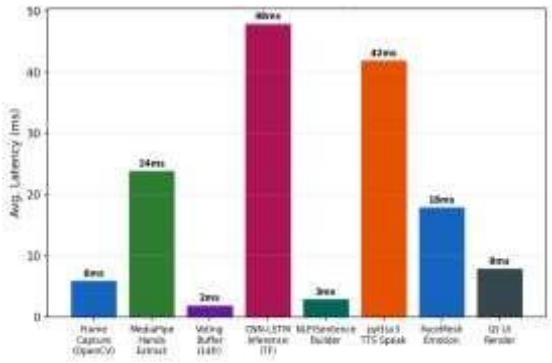

Fig. 8.5: Module-wise Processing Latency (ms) and Effective Frame Rate (FPS)

Fig. 8.6 consolidates five key metrics in a radar chart. The near-uniform polygon confirms balanced performance across Precision (91.2%), Recall (90.4%), F1-Score (90.8%), Accuracy (91.0%), and
Specificity (95.1%). Emotion detection (85.2%) shows the widest deviation, reflecting the additional challenge of landmark-ratio-based classification under variable real-world lighting conditions.

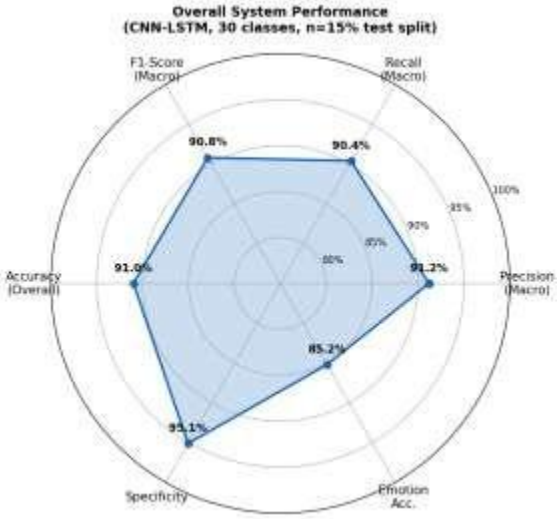

Fig. 8.6: System Performance Radar Chart (CNN- LSTM, 30 classes)

8.4 Implementation Snapshots

The following code snapshots are taken directly from the SignTalk project source files. They illustrate the three core modules: the CNN-LSTM model architecture in train_model.py, the MediaPipe landmark extraction and data augmentation pipeline in collect_data.py, and the rule-based emotion detection with stability voting in gesture_engine.py.

Code Snapshot 8.1 shows the actual build_model() function from train_model.py. The architecture uses TimeDistributed Conv1D layers (64 and 128 filters, kernel 3) applied per time-step, followed by two LSTM layers (128 and 64 units with recurrent dropout 0.1), and a Dense classifier head with BatchNormalization and Dropout(0.35/0.25). The model is compiled with Adam (lr=1e-3) and categorical cross-entropy loss.

Code Snapshot 8.1: train_model.py - build_model() (TimeDistributed CNN + LSTM, 30-class)

Code Snapshot 8.2 shows the landmark extraction pipeline from collect_data.py. MediaPipe Hands (max_num_hands=1, confidence=0.7) extracts 21 landmarks per frame into a 63-float vector [x,y,z per landmark]. Training sequences of SEQ_LEN=30 consecutive frames are collected per gesture. Augmentation doubles the training set by adding Gaussian noise (std=0.005) and random per- sample scaling (uniform 0.92-1.08).

Code Snapshot 8.2: collect_data.py - Landmark Extraction and Augmentation (30 gestures x 40 sequences)

Code Snapshot 8.3 shows the Emotion Detector.detect() method and the 14-frame stability voting from gesture_engine.py. Emotion classification uses three FaceMesh landmark ratios: MAR (mouth open), EAR (eye closure), and brow convergence gap. A deque of 20 frames provides temporal smoothing via Counter.most_common(). The _classify() method applies a 14-frame vote buffer requiring at least 50% majority agreement before emitting any gesture label, eliminating single-frame false positives.

Code Snapshot 8.3: gesture_engine.py - EmotionDetector (MAR/EAR/brow) + 14-frame Voting

System Modules and Implementation

Module 1: Data Collection (collect_data.py)

This module collects the training dataset using live webcam input via OpenCV and MediaPipe Hands (max_num_hands=1, min_detection_confidence=0.7, min_tracking_confidence=0.6). The system iterates through all 30 predefined gesture classes: HELLO, YES, NO, STOP, OK, GOOD, BAD, ME, YOU, WE, PLEASE, HELP, GO, COME, THANK YOU,WHAT, WHERE, HOW, AGAIN, FINISH, NEED, WANT, NOW, LATER, KNOW, SEE, SORRY, WAIT, GIVE, and TAKE. A 3-second
countdown is shown before each gesture begins.

When the user presses 'S', the system records 30 consecutive frames, extracting 21 hand landmarks (x, y, z) per frame into a 63-float vector using_landmarks_to_vec(). If no hand is detected, a zero vector is stored to maintain sequence length. The target is 40 sequences per gesture, saved as .npy files in data/processed/. The user can press 'R' to undo the last sequence, 'Q' to skip to the next gesture, or ESC to exit early. Gestures that already have 40 sequences are skipped automatically.

Module 2: Model Training (train_model.py)

All .npy files from data/processed/ are loaded, shape-validated as (N, 30, 63), and concatenated into a single dataset. Labels are encoded using LabelEncoder and converted to one-hot format. The data is split 85% training and 15% test using stratified splitting (random_state=42). Before training, the augment() function doubles the training set by adding Gaussian noise (std=0.005) and applying random per-sample scaling (uniform 0.92–1.08).

The CNN–LSTM model uses TimeDistributed Conv1D layers (64 and 128 filters, kernel=3) with BatchNormalization and MaxPooling for spatial feature extraction per time-step, followed by LSTM(128, return_sequences=True) and LSTM(64) for temporal modelling, and a Dense(128) → Dropout(0.35) → Dense(64) → Dropout(0.25) → Dense(30, softmax) classifier head. Training uses Adam (lr=1e-3) with three callbacks: EarlyStopping (patience=12, restores best weights), ModelCheckpoint (saves best to models/gesture_model.h5), and ReduceLROnPlateau (factor=0.4, patience=5, min_lr=1e-6). The final classification report and confusion matrix are saved to models/training_report.txt.

Module 3: Gesture Engine (gesture_engine.py)

This is the core runtime module combining rule- based classification, CNN–LSTM inference, emotion detection, and sentence building. MediaPipe Hands runs with max_num_hands=2, min_detection_confidence=0.75, and min_tracking_confidence=0.70. The
_finger_states() method extracts a 5-bit tuple (thumb, index, middle, ring, pinky) by comparing landmark positions. This state is matched againstthe exhaustive _GestureRules.TABLE, which maps all 32 possible finger combinations to gesture labels. If no rule matches, the CNN–LSTM model infers from a 30-frame landmark buffer and emits a label only when confidence exceeds 0.82.

To prevent flickering, a 14-frame voting buffer requires at least 50% majority agreement before any gesture is confirmed. The sentence state machine requires a gesture to be held continuously for HOLD_TIME=0.7 seconds before the word is added, with a COOLDOWN=1.1 seconds between successive words and a maximum of 25 words per sentence. After SPEAK_DELAY=3.5 seconds of inactivity, the sentence is passed to pyttsx3 TTS (rate=148 WPM, daemon thread) and the full state is reset.

Emotion detection runs in parallel using MediaPipe FaceMesh. Three geometric ratios are computed: Mouth Aspect Ratio (MAR), Eye Aspect Ratio (EAR), and inter-brow gap. Classification rules map these to five states: NEUTRAL, HAPPY, SURPRISED, ANGRY, and SAD. A 20-frame
history deque with majority voting provides temporal smoothing to avoid single-frame misclassifications.

Module 4: Speech-to-Text (stt.py)

The SpeechListener class provides non-blocking speech input running in a daemon thread. It probes for two backends: Google Web Speech API via SpeechRecognition as the primary, and Vosk offline model via PyAudio (16000 Hz) as the fallback. On startup, the Google backend adjusts for ambient noise. It listens with a 4-second timeout and an 8-second phrase limit. Recognised text is uppercased and delivered to the UI via a callback. Silence and unclear audio are silently ignored, and network errors trigger a 3-second retry. The listener can be started and stopped cleanly via the start() and stop() methods.

Module 5: Desktop UI (ui/desktop/main.py)

The PyQt5 desktop interface (with PySide6 fallback) features a fixed 230px dark sidebar with four navigation pages. The Live Translator page updates the camera view at approximately 30 FPS using a QTimer (33 ms interval) and displays three information cards showing the detected gesture, built sentence, and STT input. A sidebar FPS counter and mode chip (rules / CNN-LSTM) update in real time. The Speak Now button manually triggers TTS on the current sentence, and Clear resets the state.The Text to Sign page accepts typed sentences and plays back each word using a background TextToSignWorker QThread. For each word, it displays a large emoji rendered via QPainter, the hand-shape description from the GESTURE_GUIDE dictionary, and a QProgressBar showing overall progress. Each word is simultaneously spoken via pyttsx3. Playback supports pause, resume, and stop controls. The Settings page provides sliders for Gesture Hold Time and Auto-Speak Delay, and toggle checkboxes for enabling or disabling TTS, STT microphone input, and Emotion Detection at runtime.

Conclusion And Future Work

This paper presented SignTalk, a real-time, bi- directional sign language and speech translation system with integrated emotion awareness. The proposed framework combines computer vision, deep learning, natural language processing, and speech technologies into a unified communication platform. By leveraging OpenCV for video acquisition, MediaPipe for hand and facial landmark extraction, a CNN–LSTM hybrid model for continuous gesture recognition, and Google Speech-to-Text and Text-to-Speech APIs for audio interaction, the system successfully enables seamless translation between sign language and spoken language. The integration of facial and vocal emotion detection further enhances conversational naturalness by preserving contextual intent during translation.

The end-to-end implementation—from input acquisition and validation to gesture classification, speech conversion, NLP mapping, emotion analysis, and output generation—demonstrates the feasibility of deploying a real-time, inclusive communication system in practical environments. Testing results confirm stable performance, low- latency translation, and reliable bidirectional interaction. Compared to previous systems that focus primarily on unidirectional recognition or offline evaluation, SignTalk advances toward real- world applicability by integrating translation, emotion modeling, and cross-platform deployment into a single scalable architecture.

Despite its effectiveness, certain limitations remain. The current gesture vocabulary is restricted to predefined signs, and dataset diversity may affect generalization in complex scenarios. Emotion detection performance may vary under extreme

lighting or high-noise environments. Future work will focus on expanding the gesture dataset to support larger vocabularies and multiple sign languages. Advanced deep learning architectures such as Transformer-based temporal models may be explored to improve contextual understanding. Enhancing multimodal emotion fusion techniques can further improve robustness in noisy conditions. Additionally, optimizing the framework for mobile and edge deployment will increase accessibility and portability. With these enhancements, SignTalk has the potential to evolve into a fully scalable, intelligent communication assistant that promotes inclusive interaction across diverse real-world settings.

REFERENCES

[1] N. C. Camgoz, S. Hadfield, O. Koller, H. Ney, and R. Bowden, "Neural Sign Language Translation," *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[2] C. Lugaresi et al., ―MediaPipe: A Framework for Building Perception Pipelines,‖ *arXiv preprint arXiv:1906.08172*, 2019.

[3] J. Carreira and A. Zisserman, ―Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset,‖ *Proc. IEEE CVPR*, 2017.

[4] S. Hochreiter and J. Schmidhuber, ―Long Short- Term Memory,‖ *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] K. Papineni, S. Roukos, T. Ward and W. J. Zhu,
―BLEU: A Method for Automatic Evaluation of Machine Translation,‖ *Proc. ACL*, 2002.

[6] Google Cloud, ―Speech-to-Text Documentation,‖ Available: https://cloud.google.com/speech-to-text

[7] Google Cloud, ―Text-to-Speech Documentation,‖ Available: https://cloud.google.com/text-to-speech

[8] A. Duarte et al., ―How2Sign: A Large-Scale Multimodal Dataset for Continuous American Sign Language,‖ *Proc. IEEE CVPR*, 2021.