A REAL-TIME THAI SIGN LANGUAGE RECOGNITION SYSTEM USING SKELETAL KEYPOINTS AND LSTM NETWORKS

Yosvaris Pisansawanya

A Thesis in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Information Technology
Graduate Studies
Thai-Nichi Institute of Technology
Academic Year 2025

Thesis Topic	A Real-Time Thai Sign Language Recognition System Using Skeletal Keypoints and LSTM Networks
By	Yosvaris Pisansawanya
Field of Study	Information Technology
Thesis Advisor	Dr.Pramuk Boonsieng

The Graduate School of Thai-Nichi Institute of Technology has been approved and accepted as partial fulfillment of the requirement for the Master's Degree

…….. Dean of the Graduate School
(Assoc.Prof.Dr.Warakorn Srichavengsup)
Month…… Date…… Year……..……

Thesis Committees

…….. Chairman
(Dr.Pikul Vejjanugraha)

…….. Committee
(Dr.Napaphat Vichaidis)

…….. Committee
(Dr.Puwadol Sirikongtham)

…….. Advisor
(Dr.Pramuk Boonsieng)

YOSVARIS PISANSAWANYA : A REAL-TIME THAI SIGN LANGUAGE RECOGNITION SYSTEM USING SKELETAL KEYPOINTS AND LSTM NETWORKS. ADVISOR : DR.PRAMUK BOONSIENG, 85 PP.

Communication barriers remain a significant challenge for individuals who rely on sign languages, particularly in regions where language-specific recognition systems are limited. Thai Sign Language (TSL) exhibits distinctive spatial and temporal characteristics that require specialized modeling approaches for accurate and efficient real-world deployment.

This paper presents a lightweight Thai Sign Language recognition framework based on skeletal landmarks extracted using MediaPipe and modeled through Long Short-Term Memory (LSTM) networks. Instead of relying on high-dimensional raw images, the proposed approach employs a structured feature representation derived from hand and upper-body keypoints, explicitly encoding geometric relationships, finger states, and inter-hand interactions. This design improves discrimination of visually similar gestures while maintaining computational efficiency.

An end-to-end pipeline is developed, including landmark detection, feature extraction, temporal modeling, training, evaluation, and real-time inference. Experimental results show that the proposed method achieves an F1-score of 0.980 and an overall accuracy of 97.7% under a signer-dependent evaluation setting. Comparative experiments demonstrate that the proposed representation significantly outperforms raw landmark inputs under identical model configurations.

The system operates at approximately 7.31 FPS with 72 ms inference latency on a standard CPU, indicating near real-time performance. Although evaluation is limited to a signer-dependent setting, the results demonstrate a robust and efficient solution suitable for deployment on resource-constrained devices.

Graduate Studies Student's Signature……

Field of Study Information Technology Advisor's Signature……

Academic Year 2021

Acknowledgement

I would like to express my deepest gratitude to my thesis advisor, Dr. Pramuk Boonsieng, for his invaluable guidance, constructive feedback, and continuous support throughout the development of this research. His academic insight, patience, and encouragement played a crucial role in shaping this thesis and guiding me through both the technical and research challenges. Without his supervision and direction, this work would not have been successfully completed.

I would also like to sincerely thank my thesis committee members, Dr. Pikul Vejjanugraha, Dr. Napaphat Vichaidis, and Dr. Puwadol Sirikongtham, for their time, valuable suggestions, and thoughtful comments. Their expertise and constructive criticism helped improve the quality, clarity, and academic rigor of this research. I am truly grateful for their support and guidance throughout the evaluation process.

My heartfelt appreciation goes to my family for their unconditional love, understanding, and encouragement. Their constant support and belief in me provided the motivation and strength needed to overcome challenges during my studies and research journey.

I would also like to thank my colleagues and friends for their encouragement, technical discussions, and moral support during this period. Their companionship and positive energy made the research process more manageable and enjoyable.

Finally, I would like to express my sincere gratitude to the Thai-Nichi Institute of Technology (TNI) and the Faculty of Information Technology for providing the academic environment, resources, and opportunities that made this research possible. The knowledge and experience gained during my studies has been invaluable, and I am deeply thankful for the support and opportunities provided by the institute.

Yosvaris Pisansawanya

Table of Contents

Table of Contents (Continued)

List of Table

List of Figures

Chapter 1
Introduction

1.1 Background and Statement of the Problem

Communication is a fundamental aspect of human interaction, enabling access to education, employment, public services, and social participation. While spoken and written language are effective for most individuals, they present significant barriers for people who are deaf or hard of hearing, who primarily rely on sign language as their main mode of communication. Limited public understanding of sign language creates substantial communication gaps, restricting accessibility and reducing quality of life.

In Thailand, this issue is particularly significant. According to the Department of Empowerment of Persons with Disabilities, approximately 2.28 million individuals were registered as persons with disabilities as of July 2025, of which around 440,000 were identified as deaf or hard of hearing, accounting for approximately 19.27% of the total [1]. This highlights the urgent need for accessible and practical communication support systems. Figure 1.1 illustrates the distribution and scale of persons with disabilities in Thailand.

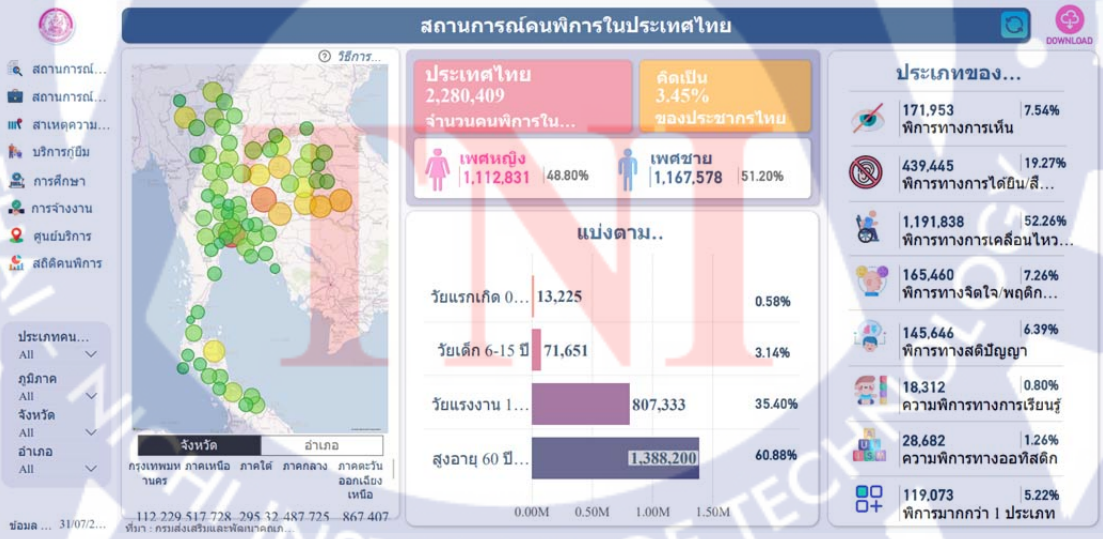


Figure 1.1 Summary of the Situation of Persons with Disabilities [1]

In recent years, sign language recognition (SLR) has advanced significantly, particularly for widely studied languages such as American Sign Language (ASL). Deep learning-based approaches, including convolutional neural networks (CNNs) and keypoint-based models, have achieved high performance under controlled conditions [2]-[4]. However, despite their strong performance, these approaches often rely on high-dimensional visual inputs or large-scale datasets, which limit their applicability in real-time and resource-constrained environments.

In contrast, research on Thai Sign Language (TSL) remains limited and underexplored. TSL presents unique linguistic and structural characteristics, including complex hand configurations, frequent two-handed coordination, and dynamic temporal motion patterns. As a result, models developed for other sign languages cannot be directly applied to TSL without performance degradation. Existing TSL studies have explored initial datasets and prototype systems [5]-[8], but many approaches focus on static gestures or simplified representations, which are insufficient for capturing the full spatio-temporal complexity of real-world communication. In addition, skeleton-based approaches have been proposed to improve the representation of hand and body motion in sign language recognition [9].

Key challenges in sign language recognition include distinguishing visually similar gestures, modeling coordinated two-hand interactions, and capturing temporal dynamics across sequences. These challenges are further amplified in real-time scenarios, where variations in lighting conditions, hand position, and user behavior introduce additional variability. Furthermore, many existing approaches rely on raw landmark coordinates, which do not explicitly encode structural relationships between joints, limiting their ability to discriminate subtle gesture differences.

To address these limitations, this study proposes a lightweight Thai Sign Language recognition framework based on skeletal keypoints and Long Short-Term Memory (LSTM) networks. Unlike conventional approaches that rely solely on raw landmark sequences, the proposed method introduces a structured feature representation that explicitly encodes geometric relationships, finger states, and inter-hand interactions, enabling improved discrimination of visually similar gestures compared to raw landmark-based approaches.

The proposed system utilizes MediaPipe to extract 21 three-dimensional keypoints for each hand, along with selected upper-body reference points to provide spatial context. These keypoints are transformed through a feature engineering pipeline that captures spatial configurations and temporal motion dynamics. The resulting features are then modeled using an LSTM-based architecture to enable sequence-based recognition. An overview of the proposed system architecture is illustrated in Figure 1.2.

By combining structured feature design with temporal modeling, the proposed approach achieves an effective balance between recognition accuracy, computational efficiency, and real-world deployability. The system is designed to operate under typical indoor conditions using standard CPU-based hardware, making it suitable for practical assistive communication applications.

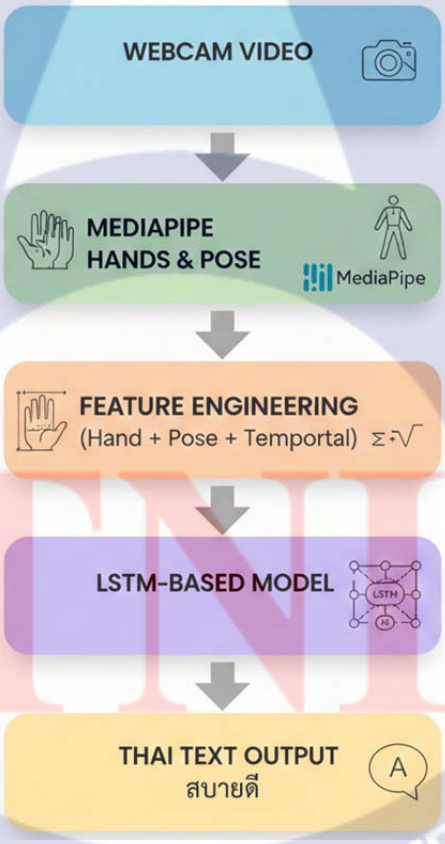


Figure 1.2 Overview of the Proposed Thai Sign Language Recognition System

1.2 Objectives of the Study

The objectives of this study are as follows:

1.2.1 To design a structured feature representation based on MediaPipe skeletal keypoints that captures geometric relationships, finger states, and inter-hand interactions for Thai Sign Language (TSL) recognition.

1.2.2 To develop a temporal modeling framework using Long Short-Term Memory (LSTM) networks for learning spatio-temporal dynamics of TSL gestures.

1.2.3 To investigate the effectiveness of the proposed feature engineering approach in improving recognition performance compared to raw landmark-based representations.

1.2.4 To evaluate the proposed system in terms of classification performance (accuracy, F1-score) and real-time capability under CPU-based deployment conditions.

1.2.5 To assess the practical feasibility of the proposed system for real-world assistive communication applications in Thai Sign Language.

1.3 Scope of Work

The scope of this study is defined as follows:

1.3.1 Data Scope:

The study utilizes a custom Thai Sign Language video dataset consisting of 30 basic words and ten numerical signs (0-9) that are commonly used in everyday communication. The selected gestures cover multiple functional categories, including feelings and emotions, greetings and blessings, adjectives and descriptors, nouns and everyday objects, actions and verbs, time expressions, transactions, transportation and location, and numerical signs. This selection aims to represent a diverse range of commonly encountered communication contexts while maintaining a manageable vocabulary size suitable for controlled experimental evaluation. Each gesture sequence is standardized to a fixed temporal length to support sequential modeling. The complete list of gesture classes and their corresponding categories is summarized in Table 1.1.

1.3.2 Technology Scope:

The system employs MediaPipe Hands for hand landmark detection and MediaPipe-based pose references for upper-body context [10]. Feature engineering and model training are implemented using TensorFlow and Keras, with an LSTM-based neural network serving as the core classification model [11], [12].

1.3.3 System Scope:

Proposed system focuses on real-time Thai Sign Language recognition from webcam video input, producing recognition results displayed as Thai text. The system is designed to operate under typical indoor conditions without specialized hardware.

Table 1.1 List of Thai Sign Language Words and Numerical Signs Used in This Study

No.	Class Name	Thai Meaning	Category	Hand Usage
1	hello	สวัสดี	Greetings & Blessings	Two-hand
2	goodLuck	โชคดี	Greetings & Blessings	One-hand
3	nervous	ประหม่า	Feeling and Emotions	Two-hand
4	iAmFine	สบายดี	Feeling and Emotions	Two-hand
5	itIsOkay	ไม่เป็นไร	Feeling and Emotions	Two-hand
6	hungry	หิว	Feeling and Emotions	One-hand
7	sorry	ขอโทษ	Feeling and Emotions	Two-hand
8	full	อิ่ม	Feeling and Emotions	Two-hand
9	thankYou	ขอบคุณ	Feeling and Emotions	Two-hand
10	cute	น่ารัก	Adjective	One-hand
11	refuse	ปฏิเสธ, ยกเลิก	Adjective / Descriptor	Two-hand
12	strong	แข็งแรง	Adjective / Descriptor	Two-hand
13	name	ชื่อ, ชื่อ-นามสกุล	Noun / Object	Two-hand
14	glasses	แว่นตา	Noun / Object	Two-hand
15	card	บัตร	Noun / Object	Two-hand
16	word	คำศัพท์	Noun / Object	Two-hand

Table 1.1 List of Thai Sign Language Words and Numerical Signs Used in This Study (Continued)

No.	Class Name	Thai Meaning	Category	Hand Usage
17	understand	เข้าใจ	Action / Verb	One-hand
18	defeat	พ่ายแพ้	Action / Verb	Two-hand
19	sick	ป่วย	Action / Verb	One-hand
20	talk	คุย	Action / Verb	Two-hand
21	keep	เก็บ	Action / Verb	Two-hand
22	wait	รอ	Action / Verb	Two-hand
23	know	ทราบ	Action / Verb	One-hand
24	shower	อาบน้ำ	Action / Verb	Two-hand
25	careful	ระมัดระวัง	Action / Verb	Two-hand
26	now	ตอนนี้, เดี๋ยวนี้	Time Expressions	Two-hand
27	money	เงิน, จ่ายเงิน	Transactions and Finance	Two-hand
28	bus	รถโดยสารประจำทาง/รถเมล์	Transportation and Location	One-hand
29	car	รถยนต์, ขับรถ	Transportation and Location	Two-hand
30	location	ตำแหน่ง, สถานที่	Transportation and Location	Two-hand
31	numberZero	เลขศูนย์	Number	One-hand
32	numberOne	เลขหนึ่ง	Number	One-hand
33	numberTwo	เลขสอง	Number	One-hand
34	numberThree	เลขสาม	Number	One-hand
35	numberFour	เลขสี่	Number	One-hand
36	numberFive	เลขห้า	Number	One-hand
37	numberSix	เลขหก	Number	One-hand
38	numberSeven	เลขเจ็ด	Number	One-hand
39	numberEight	เลขแปด	Number	One-hand
40	numberNine	เลขเก้า	Number	One-hand

The gesture categories presented in this table are defined as functional groupings to describe the general communication context of each Thai Sign Language word. These categories are intended to clarify the scope and diversity of the selected dataset and are not meant to represent a formal linguistic or grammatical classification.

1.4 Expected Contributions

The main contributions of this study are summarized as follows:

1.4.1 A lightweight feature engineering framework based on MediaPipe keypoints for Thai Sign Language recognition, capturing hand configuration, articulation, and motion dynamics.

1.4.2 A real-time gesture recognition system using LSTM-based temporal modeling, capable of operating efficiently on standard CPU-based hardware without GPU acceleration.

1.4.3 A curated Thai Sign Language dataset consisting of 40 gesture classes, including commonly used words and numerical signs, designed for controlled experimental evaluation.

1.4.4 A comprehensive evaluation of both classification performance and real-time system behavior, including accuracy, F1-score, latency, and computational efficiency.

Chapter 2
Literature Review

This chapter reviews previous studies related to sign language recognition and gesture analysis, with an emphasis on feature extraction techniques and sequence-based classification models. Particular attention is given to keypoint-based approaches that utilize keypoint detection frameworks, such as MediaPipe, and deep learning models designed for temporal data, including Long Short-Term Memory (LSTM) networks [13], [14]. These methods form the technical foundation for the proposed Thai Sign Language recognition system.

2.1 Basic Knowledge of Sign Language

Sign language is a natural and fully developed form of human communication that relies on hand shapes, finger movements, facial expressions, and body gestures rather than spoken words. It is primarily used by people who are deaf or hard of hearing, as well as by members of their families and communities. Like spoken languages, sign languages possess their own grammatical structures, vocabularies, and cultural influences [15], [16].

A common misconception is that sign language is universal. In reality, each country has its own distinct sign language, shaped by local culture and social context. Thai Sign Language (TSL) [17], for example, has been influenced historically by American Sign Language (ASL) and is reported to share approximately 52 percent similarity [18]. However, TSL maintains its own unique vocabulary, motion patterns, and semantic structures. As a result, signs with similar meanings may differ significantly across sign languages. For instance, the sign for “cute” in Thai Sign Language [19] differs from its equivalent in American Sign Language [20], and the sign for “father” in TSL is not the same as in Australian Sign Language [15]. These examples highlight that TSL should be treated as a distinct language rather than a variant of foreign sign languages.

Globally, research on sign language recognition has been dominated by studies on American and Chinese Sign Languages. In contrast, Thai Sign Language remains underrepresented in academic research. This limited coverage presents both challenges and opportunities. On one hand, the lack of large-scale datasets and standardized

benchmarks [15], [16] complicates system development and evaluation [5], [6]. On the other hand, it underscores the importance of dedicated research on TSL recognition to improve accessibility and communication support for the deaf community in Thailand [11], [17], [19]. Figure 2.1 illustrates an example comparison of the word “cute” between Thai Sign Language and American Sign Language, demonstrating the visual and structural differences between the two languages.

(a) Thai Sign Language (b) American Sign Language

Figure 2.1 Comparison of the sign for “cute” in (a) Thai Sign Language and (b) American Sign Language

2.2 MediaPipe Framework

MediaPipe is an open-source, cross-platform framework developed by Google for building and deploying real-time machine learning pipelines on edge and mobile devices [12]. The framework is specifically designed to support low-latency processing of continuous data streams, such as video and audio, making it well suited for real-time computer vision applications including face detection, pose estimation, and hand tracking. Due to its efficiency and on-device inference capability, MediaPipe has been widely adopted in gesture and sign language recognition research [9], [13], [14].

At its core, MediaPipe employs a graph-based architecture in which data flows through a sequence of modular processing units known as calculators. Each calculator performs a dedicated operation, such as object detection, region-of-interest refinement, or landmark coordinate regression. This modular design allows complex perception tasks to

be decomposed into smaller, optimized components, enabling efficient parallel execution and real-time performance on commodity hardware [10].

For hand tracking, MediaPipe Hands follows a two-stage pipeline consisting of palm detection and hand landmark estimation. In the first stage, a lightweight palm detection model identifies candidate hand regions within the input frame. These regions are then passed to a landmark regression model, which estimates 21 three-dimensional landmarks for each detected hand. The resulting landmark set provides a structured skeletal representation of hand geometry and motion, capturing joint positions and finger articulation while remaining invariant to background clutter and lighting variations [21]-[23].

Compared to raw image-based representations, landmark-based features offer significant advantages for sign language recognition. By representing hand motion as a compact set of spatial keypoints, MediaPipe reduces input dimensionality and computational cost while preserving essential geometric information. This representation is particularly suitable for downstream temporal modeling, as sequences of landmarks can be directly processed by recurrent or sequence-based neural networks without the need for additional spatial feature extraction layers [14], [23].

From a real-time perspective, MediaPipe is optimized for low-latency inference through model quantization, efficient memory management, and pipeline-level scheduling. These design choices allow hand landmark extraction to operate at interactive frame rates on standard webcams and consumer-grade hardware [10], [21], [23]-[25]. As a result, MediaPipe enables continuous tracking of one-handed and two-handed gestures, which is critical for real-world sign language recognition systems.

In the context of this research, MediaPipe serves as the front-end perception module responsible for extracting reliable skeletal keypoints from live video input. These keypoints form the spatial foundation for subsequent temporal modeling using Long Short-Term Memory (LSTM) networks, as described in Section 2.3 [26]. An example of hand landmark detection using MediaPipe Hands is illustrated in Figure 2.2.

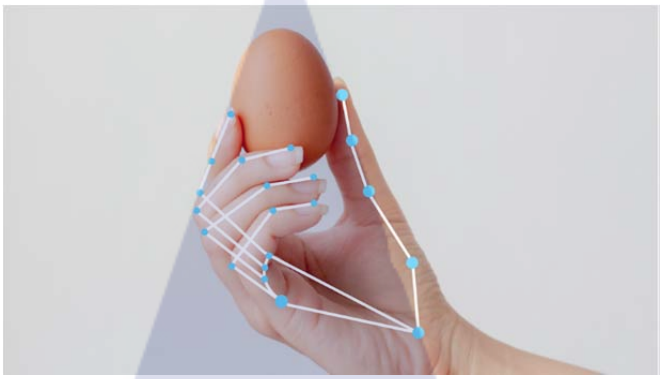


Figure 2.2 Hand Landmarks Detection [10]

2.3 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks were first introduced by Hochreiter and Schmidhuber in 1997 [26] to address a fundamental limitation of traditional Recurrent Neural Networks (RNNs), known as the vanishing gradient problem. In standard RNNs, gradients tend to diminish as sequences grow longer, making it difficult for the model to retain information from earlier time steps. This limitation significantly affects performance when learning long-term dependencies in sequential data such as speech, text, or human motion signals [26].

The key innovation of LSTM lies in its internal structure, referred to as the memory cell, which is specifically designed to preserve information over extended sequences. Each LSTM cell is regulated by three gating mechanisms that control the flow of information through the network:

• Forget Gate - determines which information from the previous cell state should be discarded as irrelevant.

• Input Gate - controls how much of the new incoming information should be stored in the memory cell.

• Output Gate - decides which portion of the stored information should be propagated forward as the hidden state for prediction or further processing.

These gates enable the LSTM to selectively retain useful information while suppressing noise, allowing the network to model long-term temporal dependencies more effectively than standard RNN architectures [26].

Figure 2.3 illustrates a chain of repeating LSTM cells, with each cell representing one step in a sequence. The key idea is how information flows through this chain [26].

• The horizontal line at the top is the Cell State (C_t). It runs straight through the entire chain, carrying information that needs to be remembered for a long time. It's what allows the LSTM to handle long-term dependencies.

• The boxes with "A" are the LSTM units themselves. Inside each unit are the three gates (forget, input, and output) that control the flow of information onto and off of the cell state.

• The circles at the bottom (X_{t-1}, X_t, X_{t+1}) represent the new data input at each step in the sequence.

• The circles at the top (H_{t-1}, H_t, H_{t+1}) are the Hidden States. These are the outputs of the LSTM unit at each step. They contain filtered information from the cell state and are passed on to the next cell.

The image shows how new data (X_t) and the previous output (H_t-1) are used by the LSTM cell to decide what to keep or forget from the memory conveyor belt (C_t-1), update it into a new memory (C_t) and then produce an output (H_t) that gets passed to the next step [26].

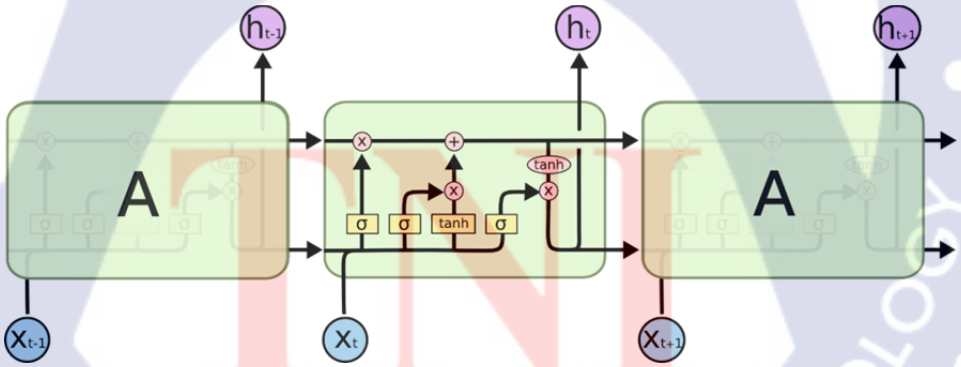


Figure 2.3 LSTM Chain Architecture [27]

In the context of sign language recognition, LSTM networks are particularly well suited due to the inherently sequential nature of sign gestures. A single sign is not defined solely by a static hand shape but by a continuous sequence of movements, transitions, and temporal dependencies between frames. By processing sequences of

skeletal keypoints extracted from consecutive video frames, LSTM models can effectively capture both spatial configurations and motion dynamics of sign gestures [11]-[14], [23], [24], [26].

Previous studies have demonstrated that LSTM-based architectures achieve reliable performance for real-time sign language recognition when combined with structured landmark representations, such as those provided by MediaPipe [10], [21], [23]-[25]. These findings support the selection of LSTM networks as the core temporal modeling component in this research, enabling accurate and stable recognition of Thai Sign Language gestures under real-time conditions.

2.4 Related Works

This section reviews prior research related to sign language recognition, with a focus on the evolution of methodologies, the use of MediaPipe keypoint representations, and the adoption of deep learning models for temporal modeling. The reviewed works are organized to highlight key trends, strengths, and limitations that inform the motivation of this research, particularly in terms of robustness, scalability, and real-time deployment challenges [15], [16].

Early research on sign language recognition primarily focused on isolated and static gestures, treating each sign as a single image rather than a temporal sequence. Convolutional Neural Networks (CNNs) were commonly employed in this stage due to their strong performance in spatial feature extraction. Several studies demonstrated that CNN-based approaches could achieve high accuracy for finger-spelling or numerical sign recognition under controlled conditions, particularly when background complexity was limited and gestures were clearly segmented [28]-[31]. However, these image-based methods exhibited limitations when applied to dynamic signs, where motion and temporal transitions play a critical role in conveying meaning [15].

To overcome the limitations of static analysis, subsequent studies shifted toward sequence-based models capable of learning temporal dependencies. Recurrent architectures, particularly Long Short-Term Memory (LSTM) networks, became a popular choice for modeling sign language as a sequence of movements rather than independent frames. Representative systems, such as SLRNet, demonstrated that stacking landmark sequences over fixed temporal windows and processing them with LSTM

networks could achieve reliable performance while maintaining real-time capability on standard hardware [3]. Similar approaches combining CNN-based spatial feature extraction with LSTM-based temporal modeling were also reported to improve recognition accuracy for dynamic gestures [29], [32].

The availability of pre-trained landmark extraction frameworks further accelerated progress in sign language recognition. MediaPipe, in particular, has been widely adopted due to its ability to provide reliable hand, pose, and facial landmarks in real time without the need for training custom detectors [10]. By replacing raw image input with structured keypoint-based representations, MediaPipe-based approaches significantly reduce computational complexity and enable efficient downstream temporal modeling. Numerous studies have successfully integrated MediaPipe landmarks with LSTM or BiLSTM architectures for real-time sign language recognition across different languages, including American, Norwegian, and Indian Sign Languages [23]-[25].

Within the context of Thai Sign Language (TSL), existing research has applied both static and dynamic recognition approaches with promising but limited results. Early work on Thai finger-spelling and numerical sign recognition demonstrated that CNN-based models can achieve high accuracy on isolated gestures, particularly when hand regions are cropped or segmented [30]. More recent studies employed MediaPipe Holistic landmarks combined with LSTM or BiLSTM models to translate Thai signs into text, achieving improved performance on experimental datasets [11], [30]. These studies confirmed that temporal modeling and multi-modal landmark representations are beneficial for TSL recognition.

Despite these advances, several challenges remain unresolved in existing TSL research. First, many studies rely on small or single-signer datasets, which limits generalization and leads to performance degradation when systems are deployed in real-time environments [11], [13]. Second, a significant portion of prior work focuses on isolated signs or finger-spelling, while continuous or real-time recognition remains comparatively underexplored [15]. Third, handling two-handed gestures and self-occlusion continues to be a major challenge, as landmark detectors may struggle when one hand overlaps or interacts with the other [8], [9]. Additionally, several studies report discrepancies between offline evaluation results and real-time performance, highlighting the need for evaluation under realistic operating conditions [25].

Recent research trends have begun exploring more advanced architectures, such as graph-based models, spatial–temporal graph convolutional networks (ST-GCN), and transformer-based frameworks for skeleton-based sign language recognition [32]-[36], [37]-[39]. While these methods demonstrate strong performance on large-scale datasets, they often require higher computational resources and longer training times, making them less suitable for lightweight or real-time applications on consumer-grade hardware [33], [36].

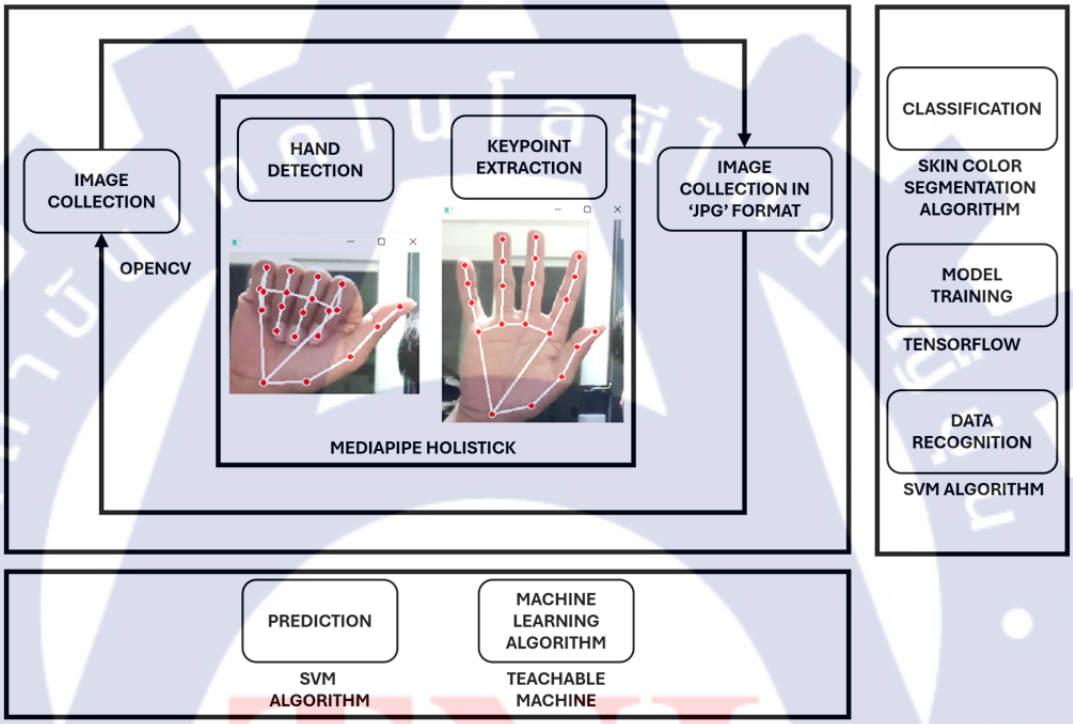


Figure 2.4 SLRNet System Architecture

Studies within the Thai context have applied these modern techniques and revealed important insights:

• Hand-focused representations have been shown to improve generalization performance. A study on Thai numerical sign recognition reported that a hand-cropped VGG-LSTM model achieved 81.25% accuracy, indicating the effectiveness of focusing on hand regions [5].

• Temporal modeling and multi-modal keypoint representations significantly improve recognition accuracy. Studies using MediaPipe Holistic combined with LSTM and BiLSTM models reported accuracies of approximately 0.83 and 0.91, respectively [6], [11].

• Occlusion and two-hand interaction remain major challenges. A MediaPipe-based study on Thai finger-spelling reported strong performance for single-hand gestures (~84.6%) but significantly lower accuracy (~23.7%) under occlusion conditions [8].

Real-time performance remains more challenging than offline evaluation. A recent study on continuous Thai Sign Language recognition reported that while an LSTM model achieved 100% accuracy on a test dataset, performance decreased to approximately 86% under real-time conditions [40]-[42]. This discrepancy indicates potential overfitting in single-signer datasets and highlights the importance of evaluation under realistic deployment scenarios. Figure 2.5 presents a comparison of model evaluation results.

Models	Model accuracy	Real-time test accuracy
RNN	1.00	0.64
Bi-RNN	0.87	0.50
LSTM	**1.00**	**0.86**
Bi-LSTM	0.06	N/A
FNN+LSTM	0.87	0.56

The bold front indicates the best model

Figure 2.5 A Comparison of Model Evaluation

Building on these findings, the present study integrates MediaPipe-based keypoint extraction with LSTM-based temporal modeling to effectively capture both spatial hand configurations and motion dynamics. This approach aims to provide a balance between recognition accuracy and computational efficiency for real-time Thai Sign Language recognition.

A comparative summary of representative approaches is presented in Table 2.1, highlighting differences in methodology, performance, and deployment constraints.

Table 2.1 Related Works Comparison

Ref.	Research Focus	Methodology	Key Findings & Performance Metrics	Limitations & Implications for This Research
[5]	Thai Sign Language Numerical Sign Recognition	CNN / CNN-LSTM on hand-cropped videos	Best out-of-sample accuracy 81.25%, F1-score 85.21%; hand-cropping improves performance	Restricted to numerical signs; performance degrades outside controlled settings, motivating vocabulary-rich TSL recognition
[6],[11]	Thai SL detection and conversion	MediaPipe Holistic + LSTM	Reported ~0.83 accuracy with Thai text output	Accuracy drops in real-time and models are computationally heavy, motivating a lightweight CPU-oriented approach
[7]	Thai finger-spelling recognition	CNN with optical flow and semantic segmentation	Achieved ~88–92% accuracy; ambiguity increases with multi-stroke gestures	Limited to finger-spelling; sensitive to stroke segmentation, highlighting need for more robust feature representations
[8]	MediaPipe Hands for Thai finger-spelling	Landmark extraction + ANN / rule-based methods	Single-hand accuracy 84.57%; failures under occlusion and handedness variation	Landmark instability motivates normalization and robustness enhancement adopted in this research
[11]	Real-time Thai SL translation	MediaPipe Holistic → LSTM (30-frame sequences)	Achieved 100% test accuracy but ~86% in real-time (single signer)	Overfitting and limited signer diversity highlight need for generalizable, multi-signer systems

Table 2.1 Related Works Comparison (Continued)

Ref.	Research Focus	Methodology	Key Findings & Performance Metrics	Limitations & Implications for This Research
[5], [17]	Early Thai SL recognition	DNN / landmark clustering	Demonstrated feasibility on small Thai datasets	Limited vocabulary and modest accuracy motivate scalable and feature-rich TSL systems
[2], [4]	Real-time isolated SLR	MediaPipe hand landmarks + CNN/LSTM	Demonstrated real-time feasibility with good accuracy under controlled environments	Typically limited vocabulary and unclear deployment constraints, motivating CPU-only practical systems
[13], [23]	Landmark-based SLR (non-Thai)	MediaPipe landmarks + LSTM	Achieved ~95% accuracy on numeric gestures	The approach is limited to numerical signs and, although effective for skeletal modeling, underscores the need for larger and more diverse vocabularies.
[30], [31], [39]	Feature engineering for SLR	Geometric and phonological skeletal features	Improved discrimination between visually similar signs	Feature complexity must be balanced with efficiency, guiding selective feature design in this research
[34], [35]	Holistic and multi-part landmark fusion	Hand + pose landmark integration	Improved recognition of similar gestures	Increased computational cost motivates selective use of arm cues while preserving CPU performance

Table 2.1 Related Works Comparison (Continued)

Ref.	Research Focus	Methodology	Key Findings & Performance Metrics	Limitations & Implications for This Research
[33], [34], [36]	Transformer-based SLR	Skeleton-based Transformer models	Reported very high benchmark accuracy (up to ~99% on LSA-64)	GPU-oriented and computationally intensive, contrasting with the CPU-only design of this research
[40], [42]	Continuous SLR	Continuous skeleton-based or Transformer frameworks	Enable sentence-level recognition	Require segmentation and large datasets; considered future work rather than focus of this research
[15], [16]	Review and survey studies	Literature synthesis	Identified challenges in robustness, scalability, and deployment	Reinforces the need for accurate, vocabulary-rich, and deployable TSL systems

A comparative synthesis of representative sign language recognition studies, including their modeling approaches, performance characteristics, and deployment considerations, is presented in Table 2.1. Rather than comparing individual systems solely by recognition accuracy, the table highlights key trade-offs among vocabulary coverage, computational complexity, and real-time feasibility across existing approaches [43]-[45].

From the reviewed literature, it can be observed that MediaPipe-based skeletal landmark extraction combined with LSTM-based temporal modeling provides a favorable balance between recognition accuracy and computational efficiency [46], [47]. This approach is particularly suitable for real-time applications on resource-constrained platforms. However, existing applications to Thai Sign Language are often limited by small vocabularies, reduced robustness under real-time conditions, and a lack of emphasis on CPU-only deployment [48], [49]. These limitations restrict the practical applicability of current systems in everyday assistive communication scenarios.

Problems Statement

Based on the reviewed literature, it is evident that although MediaPipe-based skeletal landmark extraction and LSTM-based temporal modeling have demonstrated strong potential for real-time sign language recognition, their application to Thai Sign Language remains limited in practical scope [46], [47]. Existing TSL studies predominantly focus on restricted vocabularies, finger-spelling, or controlled experimental settings, with limited evaluation of large-vocabulary recognition under real-time conditions [48].

Moreover, while recent transformer-based and graph-based models report high recognition accuracy on benchmark datasets, their reliance on computationally intensive architectures limits their suitability for real-time deployment on standard CPU-based systems [43]. This creates a gap between high-accuracy research models and practical assistive technologies that must operate reliably without specialized hardware such as GPUs.

Consequently, there is a clear need for a Thai Sign Language recognition framework that (1) supports a broader vocabulary of commonly used signs, (2) maintains high recognition accuracy under real-time conditions, (3) operates efficiently on CPU-only platforms, and (4) is evaluated in settings representative of daily communication. This research addresses these gaps by integrating MediaPipe-based skeletal feature extraction with LSTM-based temporal modeling to develop a practical, accurate, and deployable real-time Thai Sign Language recognition system.

Proposed Approach

To address the identified limitations, this study proposes a lightweight yet effective Thai Sign Language recognition framework that combines MediaPipe-based skeletal landmark extraction with LSTM-based temporal sequence modeling [46], [47]. The proposed approach focuses on extracting discriminative hand and finger keypoint features from video input and modeling their temporal dynamics using a recurrent neural network architecture optimized for sequential data. Instead of relying on computationally expensive transformer-based or graph-based models, the framework emphasizes efficient feature engineering and temporal learning to achieve a balance between recognition accuracy and computational efficiency [43].

By leveraging normalized skeletal keypoints and carefully designed temporal feature representations, the proposed method enables real-time inference on CPU-only platforms while supporting a broader vocabulary of commonly used Thai Sign Language signs [48], [49]. The system is evaluated under realistic conditions, including variations in signer behavior, hand distance, and lighting, to ensure its suitability for practical deployment in assistive communication scenarios.

Chapter 3
Research Methodology

This chapter presents the methodology employed in the development of the proposed real-time Thai Sign Language (TSL) recognition system. The primary objective of this methodology is to design a practical and robust framework capable of accurately recognizing dynamic Thai Sign Language gestures while operating efficiently under real-time constraints.

Based on the research gaps identified in Chapter 2, this study adopts a skeleton-based recognition approach that leverages hand landmark extraction and temporal sequence modeling. The proposed system is designed to capture essential spatial and temporal characteristics of sign gestures while maintaining low computational complexity, making it suitable for real-time deployment on standard hardware. To achieve this, the framework integrates MediaPipe Hands for skeletal landmark extraction and a Long Short-Term Memory (LSTM) neural network for temporal modeling and classification.

This chapter details the system architecture, data processing pipeline, feature representation, model design, training procedure, and real-time inference strategy used in this research. The overall workflow of the proposed system is illustrated in Figure 3.1, followed by a detailed explanation of each processing stage in the subsequent sections.

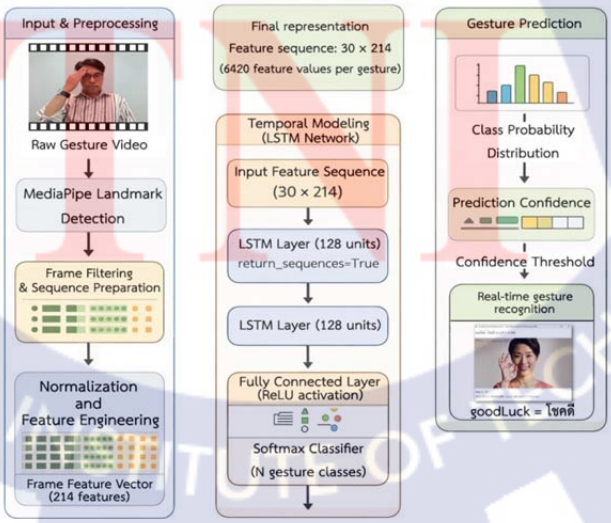


Figure 3.1 Thai Sign Language Recognition using LSTM Workflow

3.1 Video Input

Video input constitutes the primary data source for both the offline training phase and the real-time operation of the proposed Thai Sign Language (TSL) recognition system. In this study, video data are obtained from two complementary sources: pre-recorded gesture videos used for model training and evaluation, and live video streams captured from a webcam for real-time inference. This dual-input design enables the system to be trained under controlled conditions while being evaluated in realistic deployment scenarios.

For the training phase, a custom Thai Sign Language (TSL) video dataset is constructed using pre-recorded gesture sessions. Each video corresponds to a specific TSL word or numerical sign and is performed by multiple signers. To provide a clearer overview of the dataset characteristics and ensure reproducibility, the overall dataset composition is summarized in Table 3.1.

Table 3.1 Dataset Composition Summary

Parameter	Description
Number of gesture classes	40 Thai Sign Language gestures
Samples per class	30 videos per class
Number of signers	3 signers
Total video samples	1,200 videos
Average samples per class	5 samples
Video resolution	1,280 × 720 pixels
Frame rate	30 FPS
Gesture type coverage	One-hand and two-hand gestures

As summarized in Table 3.1, the dataset comprises 40 Thai Sign Language gesture classes collected from three signers, resulting in a total of 1,200 video samples with balanced coverage of one-hand and two-hand gestures.

To enhance data diversity and improve generalization, recordings are conducted under varying lighting conditions, hand-to-camera distances, and minor viewpoint changes. This strategy is consistent with prior Thai Sign Language studies, which emphasize the importance of signer and environmental variability in reducing overfitting

and improving robustness [30], [41], [44], [50]. Representative examples of real-time video captures and pre-recorded gesture samples are shown in Figures 3.2 and 3.3, respectively.

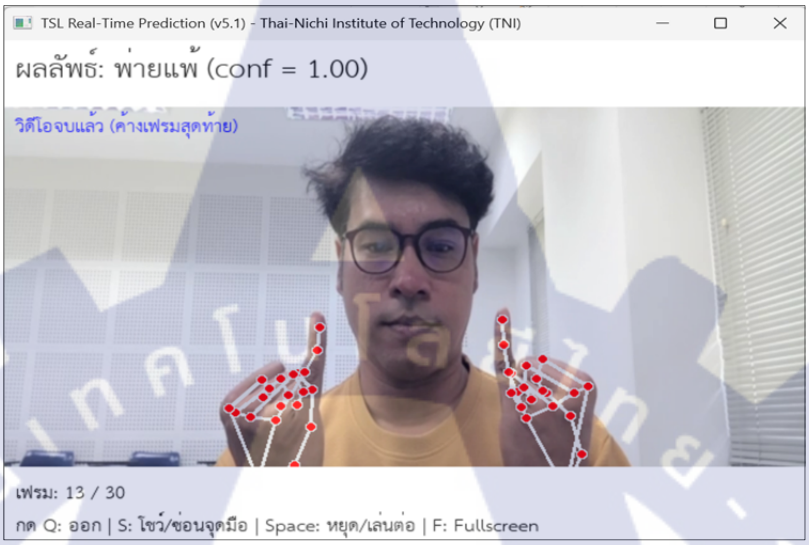


Figure 3.2 Real-time video captures

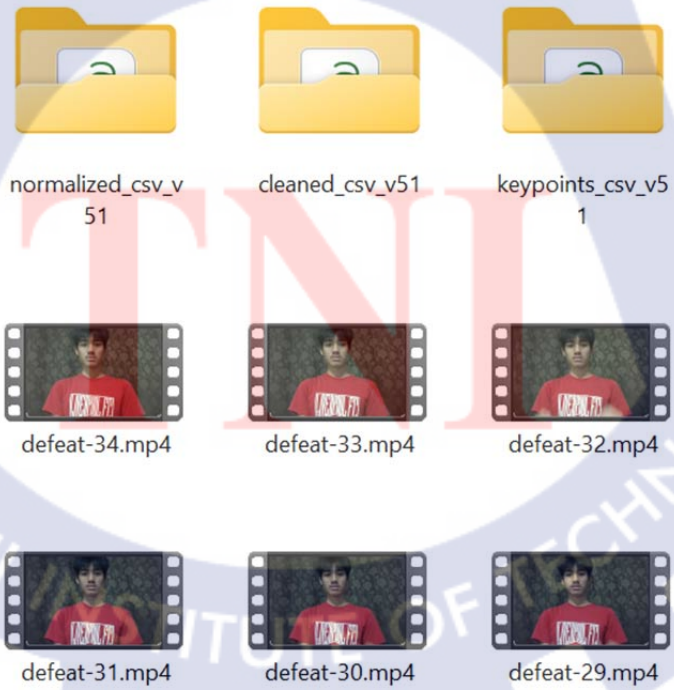


Figure 3.3 Pre-recorded sessions video input

All training videos are recorded using a consistent resolution and frame rate to preserve temporal uniformity across samples. Maintaining a stable frame rate is particularly important for sequence-based models such as LSTM, as irregular temporal spacing may introduce inconsistent motion patterns that degrade temporal learning performance [37], [42]. During dataset preparation, videos are processed to ensure compatibility with fixed-length temporal segmentation used in later stages of the pipeline.

To support efficient processing and reproducibility, the dataset is organized using a class-based directory structure, where each folder represents a single Thai Sign Language gesture. Non-relevant files, such as system assets or auxiliary resources, are excluded from the dataset directories. This structured organization facilitates automated skeletal keypoint extraction, label assignment, and batch processing during model training, as commonly adopted in recent sign language recognition frameworks [33]-[35].

In the real-time inference phase, video input is captured directly from a webcam and processed frame by frame without offline buffering or prior segmentation. This design choice prioritizes low-latency operation and allows the system to respond immediately to dynamic hand movements. Unlike pre-recorded training data, real-time input may include idle periods or frames where no hands are visible. Such frames are handled dynamically during processing to prevent invalid or outdated motion information from being incorporated into temporal feature sequences, a challenge frequently reported in real-time sign language recognition systems [41], [45], [47].

Overall, the video input design establishes a unified interface for both offline model training and online recognition. By combining structured pre-recorded datasets with continuous webcam-based input, the proposed system aligns with practical deployment requirements for real-time assistive communication while maintaining consistency with established methodologies in skeleton-based sign language recognition research [37], [43], [47].

3.2 Hand Landmark Detection and Keypoint Extraction using MediaPipe

Following video frame acquisition, the next stage of the proposed system is hand landmark detection and keypoint extraction. This stage aims to transform raw visual input into a structured skeletal representation suitable for temporal modeling. In this

research, MediaPipe Hands is employed due to its accuracy, efficiency, and suitability for real-time sign language recognition.

MediaPipe is built upon a graph-based processing architecture in which multimedia data flows through interconnected processing units known as Calculators. Each Calculator performs a specific task, such as palm detection, region-of-interest (ROI) refinement, hand landmark estimation, or coordinate normalization. This modular pipeline design enables efficient parallel processing and supports low-latency execution, making MediaPipe well suited for real-time applications [34], [35].

3.2.1 MediaPipe Hands Detection Pipeline

MediaPipe Hands operates using a two-stage deep learning pipeline:

(1) Palm Detection:

In the first stage, a lightweight palm detection model identifies the approximate location of a hand within the input frame. By focusing on palm regions rather than individual fingers, this detector achieves robust performance under partial occlusion, self-overlapping fingers, and varying illumination conditions.

(2) Hand Landmark Estimation:

Once a palm region is detected, the second stage of the MediaPipe Hands pipeline performs detailed hand landmark estimation. In this stage, a dedicated deep learning model predicts the positions of 21 anatomical landmarks for each detected hand, capturing key joints of the wrist, fingers, and fingertips. These landmarks collectively form a skeletal representation of hand posture and configuration, as illustrated in [Figure 3.4]. The landmark definition follows the standard hand model proposed in the MediaPipe Hands framework [34], [35].

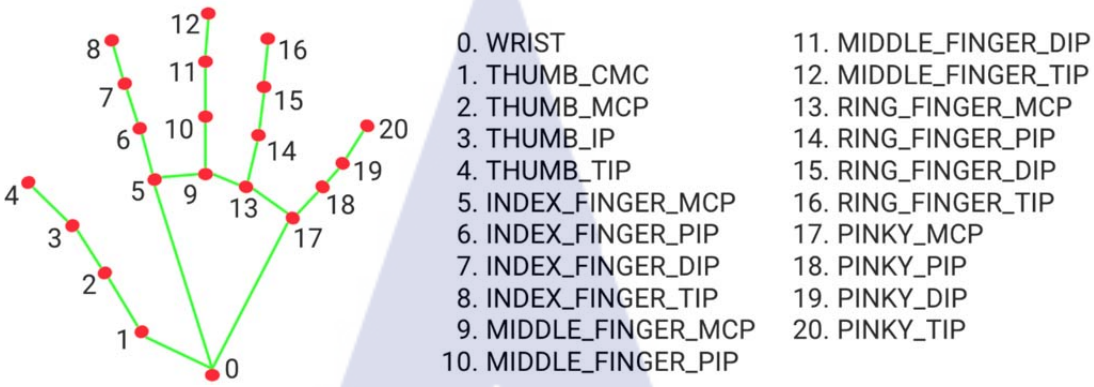


Figure 3.4 Hand Skeleton with 21 Landmarks Defined in MediaPipe [10]

The definitions and anatomical descriptions of all 21 hand landmarks are summarized in Table 3.2.

Table 3.2 Media Pipe Hand Landmark Description

Index	Landmark Name	Description
0	WRIST	Wrist (main reference point)
1	THUMB_CMC	Thumb base (carpometacarpal joint)
2	THUMB_MCP	Thumb knuckle (metacarpophalangeal joint)
3	THUMB_IP	Thumb middle joint (interphalangeal joint)
4	THUMB_TIP	Tip of the thumb
5	INDEX_MCP	Index finger knuckle (MCP joint)
6	INDEX_PIP	Index finger middle joint (PIP joint)
7	INDEX_DIP	Index finger distal joint (DIP joint)
8	INDEX_TIP	Tip of the index finger
9	MIDDLE_MCP	Middle finger knuckle (MCP joint)
10	MIDDLE_PIP	Middle finger middle joint (PIP joint)
11	MIDDLE_DIP	Middle finger distal joint (DIP joint)
12	MIDDLE_TIP	Tip of the middle finger
13	RING_MCP	Ring finger knuckle (MCP joint)
14	RING_PIP	Ring finger middle joint (PIP joint)

Table 3.2 Media Pipe Hand Landmark Description (Continued)

Index	Landmark Name	Description
15	RING_DIP	Ring finger distal joint (DIP joint)
16	RING_TIP	Tip of the ring finger
17	PINKY_MCP	Little finger knuckle (MCP joint)
18	PINKY_PIP	Little finger middle joint (PIP joint)
19	PINKY_DIP	Little finger distal joint (DIP joint)
20	PINKY_TIP	Tip of the little finger

Table 3.2 lists the 21 hand landmarks detected by MediaPipe Hands, covering the wrist and all finger joints, which together provide a complete skeletal representation of hand posture for feature extraction and temporal modeling.

The estimated landmarks provide a structured and compact description of hand geometry that is invariant to scale and moderately robust to changes in viewpoint. Each landmark corresponds to a specific anatomical joint, enabling fine-grained representation of finger articulation and hand orientation. Such detailed skeletal information is particularly important for sign language recognition, where many gestures differ only in subtle finger configurations or relative joint positions [13], [31], [43].

Figure 3.5 shows that the landmark estimation stage operates on a refined region of interest produced by the palm detection module. By restricting landmark prediction to the detected hand region, the system reduces background interference and improves both detection accuracy and computational efficiency. This two-stage design—consisting of palm detection followed by landmark estimation—has been shown to support stable and reliable hand tracking under real-time constraints, even in the presence of partial occlusion, rapid hand motion, or varying illumination conditions [34], [35].

Figure 3.5 MediaPipe Hands Processing Pipeline (Palm Detection → Landmark Estimation)

3.2.2 Landmark Representation and Dimensionality

Each detected landmark P_j is represented using three normalized coordinates, as illustrated in Figure 3.4.:

$$P_j = (x_j, y_j, z_j) \tag{3.1}$$

- x: horizontal position relative to the image width
- y: vertical position relative to the image height
- z: relative depth value representing distance from the camera

As a result, each hand produces $21 \times 3 = 63$ numerical values per frame. When both hands are detected simultaneously, the total dimensionality becomes 126 values per frame. This compact landmark-based representation significantly reduces input dimensionality compared to raw image data while preserving essential spatial and structural information required for sign language recognition.

Compared to raw image-based representations, skeletal keypoints significantly reduce input dimensionality while preserving essential spatial relationships required for sign language recognition. This efficiency makes landmark-based features particularly suitable for sequence-based deep learning models and real-time applications [34], [35], [43].

Frames in which no hands are detected are explicitly identified and handled by the system to prevent invalid or missing data from propagating into subsequent temporal modeling stages. This handling is particularly important in real-time operation, where users may pause between gestures or temporarily move their hands outside the camera view.

3.2.3 Suitability for Real-Time Sign Language Recognition

The landmark-based output of MediaPipe Hands is well suited for real-time Thai Sign Language recognition. Many TSL gestures rely heavily on finger articulation, hand orientation, and relative hand motion rather than full-body posture. By focusing exclusively on hand landmarks, the system reduces computational overhead and

minimizes sensitivity to background noise, while still capturing the most discriminative features for gesture recognition [30], [31], [43].

Furthermore, MediaPipe Hands maintains temporal consistency by tracking detected hands across consecutive frames, reducing jitter and improving stability in landmark trajectories. This temporal coherence is critical for sequence-based models such as LSTM, which rely on smooth and meaningful motion patterns over time [37], [42].

The extracted hand landmarks from this stage serve as the primary input for subsequent feature construction and normalization, as described in the next section.

3.3 Feature Construction and Normalization

After hand landmarks are extracted for each video frame, the next stage of the proposed system focuses on feature construction and normalization. The objective of this stage is to transform raw landmark coordinates into structured, consistent, and temporally aligned input sequences that are suitable for input to a Long Short-Term Memory (LSTM) network. Emphasis is placed on ensuring uniform input dimensionality, temporal stability, and robustness under real-time operating conditions.

At the frame level, spatial features are constructed by concatenating hand and pose landmarks into a unified feature vector. Each detected hand contributes 21 landmarks represented by three-dimensional coordinates (x, y, z), while selected upper-body pose landmarks are included as spatial reference points to stabilize hand motion interpretation. The resulting frame-level input structure is summarized in Table 3.3.

Table 3.3 Frame-level Input Feature Structure (147 features)

Feature Component	**Landmarks**	**Coordinates**	**Features per Frame**
Left Hand	21	(x, y, z)	63
Right Hand	21	(x, y, z)	63
Pose Reference (nose + arms)	7	(x, y, z)	21
Total (Input Features)	-	-	147
Label (supervised learning)	-	-	1

This table summarizes the raw frame-level input structure prior to feature engineering. Higher-level geometric and articulation-based features are derived in Section 3.4.

At the frame level, each detected hand landmark is represented by normalized three-dimensional coordinates, forming a compact skeletal description of hand posture and motion. Frames in which no hands are detected are explicitly identified and excluded to prevent invalid or misleading spatial information from influencing temporal modeling. This handling is particularly important in real-time sign language recognition scenarios, where idle periods, brief occlusions, or temporary loss of hand visibility frequently occur.

To ensure uniform temporal input across all gesture samples, variable-length video clips are standardized to a fixed sequence length. After removing invalid frames, longer gesture sequences are downsampled when necessary to reduce redundancy, while shorter sequences are padded by repeating the last valid frame. In this study, all gesture sequences are resized to a fixed length of 30 frames, providing a balance between capturing sufficient motion dynamics and maintaining computational efficiency. Similar temporal normalization strategies have been widely adopted in prior sign language recognition research [37], [41], [45], [47].

The overall process of temporal sequence construction and resizing-from raw video frames, through frame filtering, downsampling, and padding, to fixed-length feature sequences which is illustrated in Figure 3.6.

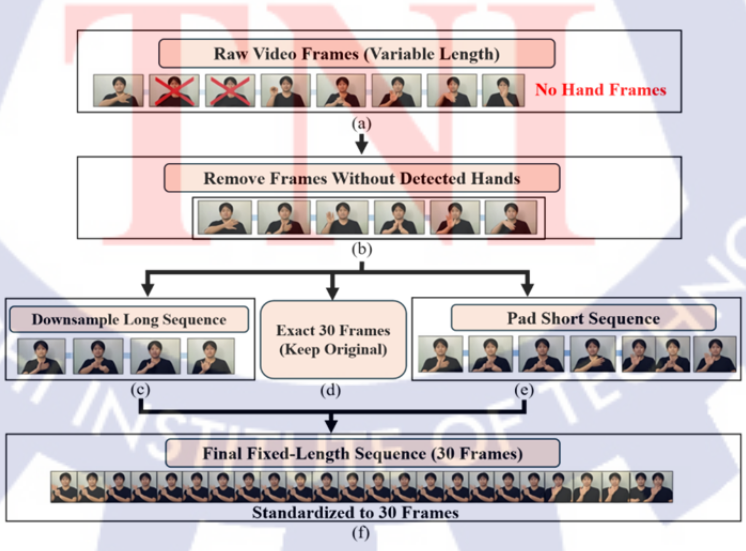


Figure 3.6 Temporal sequence construction and resizing

This combination of frame filtering, temporal resizing, and normalization ensures that all input sequences share a consistent dimensional structure. As a result, the LSTM model can focus on learning discriminative temporal patterns of Thai Sign Language gestures rather than adapting to variable input lengths or noisy spatial representations. The detailed procedures for frame-level feature construction, handling of missing frames, temporal resizing, and normalization are described in the following subsections.

3.3.1 Frame-level Feature Construction

At the frame level, each video frame containing detected hands is converted into a structured numerical feature vector. As illustrated in Figure 3.6 (a), raw input videos may contain frames in which no hands are visible; therefore, only frames with detected hands are retained for subsequent feature extraction. This process transforms raw skeletal landmark outputs from MediaPipe into a consistent representation suitable for temporal modeling with a Long Short-Term Memory (LSTM) network.

For each detected hand, MediaPipe Hands provides 21 anatomical landmarks, where each landmark is represented by three normalized spatial coordinates (x, y, z). The x and y coordinates denote the relative horizontal and vertical positions within the image frame, while the z coordinate represents relative depth information. As a result, each hand contributes 63 numerical features per frame. When both hands are detected simultaneously, the combined hand representation consists of 126 features per frame.

In addition to hand landmarks, a subset of upper-body pose landmarks is incorporated to provide spatial reference and contextual stability for hand movement interpretation. Specifically, the nose and selected arm joints (left and right shoulders, elbows, and wrists) are included as reference points. These pose landmarks contribute an additional 7 landmarks, each represented by three coordinates, resulting in 21 pose-related features per frame. This design helps stabilize hand trajectories relative to the upper body and improves robustness against camera movement and variations in signer position, as suggested in prior skeleton-based sign language recognition studies [37], [41], [43].

All extracted landmarks are concatenated in a fixed and consistent order to form a single frame-level feature vector. Excluding the class label, each valid frame is therefore represented by a total of 147 numerical features, comprising left-hand landmarks,

right-hand landmarks, and pose reference landmarks. When stored in tabular form for supervised learning, an additional label column is appended, resulting in 148 columns per frame.

Frames in which one or both hands are not detected are handled explicitly to maintain feature consistency. Missing hand landmarks are represented using predefined placeholder values, allowing the feature vector dimensionality to remain constant across frames. Frames with no detected hands are flagged for removal or special handling in subsequent processing stages, as described in the following subsection.

This frame-level feature construction step establishes a unified and structured representation of spatial hand and pose information. By standardizing the feature format at the frame level, the system ensures compatibility with subsequent temporal resizing, normalization, and sequence modeling stages, which are essential for effective LSTM-based Thai Sign Language recognition.

3.3.2 Handling Missing and Invalid Frames

In practical sign language recognition scenarios, especially under real-time conditions, not all video frames contain valid or complete hand detections. Such situations may occur due to temporary occlusion, rapid hand movement, idle pauses between gestures, or the signer moving outside the camera's field of view. If not handled properly, these frames can introduce noise and degrade the performance of temporal models.

In this study, frames in which no hands are detected are explicitly identified and excluded from sequence construction, as shown in Figure 3.6 (b). This prevents invalid spatial information from contaminating the temporal feature sequence. For frames where only one hand is detected, the missing hand landmarks are replaced with predefined placeholder values, ensuring that the dimensionality of the frame-level feature vector remains consistent across all frames. This strategy allows the system to support both one-handed and two-handed Thai Sign Language gestures within a unified feature representation.

By distinguishing between invalid frames (no detected hands) and partially valid frames (one detected hand), the proposed system maintains temporal continuity while minimizing the influence of unreliable spatial data. Similar handling

strategies have been reported as essential for robust real-time sign language recognition systems operating under unconstrained conditions [34], [42].

3.3.3 Temporal Sequence Resizing

Gesture sequences extracted from video input naturally vary in length due to differences in signer speed, gesture complexity, and recording conditions. However, sequence-based models such as LSTM require fixed-length input for efficient batch training and stable inference [29], [37], [41], [45].

To address this constraint, all gesture sequences in this study are standardized to a fixed temporal length. Initially, frames without detected hands are removed as described in the previous subsection. For gesture clips that exceed the target sequence length, temporal downsampling is applied to reduce redundancy while preserving overall motion characteristics, as shown in Figure 3.6 (c). Conversely, shorter sequences are padded by repeating the last valid frame until the target length is reached, as shown in Figure 3.6 (e).

In this research, a fixed sequence length of 30 frames is adopted. This value represents a trade-off between capturing sufficient temporal dynamics of Thai Sign Language gestures and maintaining computational efficiency for real-time operation. Prior studies on sign language recognition have similarly adopted fixed sequence lengths in the range of 20–40 frames for LSTM-based modeling, demonstrating effective temporal learning with manageable complexity.

This temporal cleaning and resizing process ensures that all gesture samples are represented by sequences of uniform length, enabling consistent input to the LSTM network during both training and real-time inference.

3.3.4 Final Feature Vector Structure

After frame-level construction, missing frame handling, temporal resizing, and normalization, each gesture sample is represented as a fixed-size feature tensor suitable for LSTM-based classification, as shown in Figure 3.6 (f). Specifically, each gesture sequence is structured as a three-dimensional tensor of shape:

$$(T, F) = (30,147)$$

where $T = 30$ represents the fixed temporal length, and $F = 147$ denotes the number of numerical features per frame, comprising hand landmarks and pose reference landmarks. For supervised learning and dataset storage, an additional label column is appended, resulting in 148 columns per frame in the tabular representation, as mentioned in Table 3.3.

This standardized feature structure enables efficient batch training, consistent model evaluation, and seamless deployment in real-time inference. By enforcing uniform dimensionality at both the frame and sequence levels, the proposed methodology ensures that the LSTM model can focus on learning discriminative temporal patterns inherent in Thai Sign Language gestures, rather than adapting to inconsistent input formats.

3.4 Feature Engineering

After sequence cleaning, temporal resizing, and normalization described in the previous section, this stage focuses on feature engineering, where MediaPipe keypoints are transformed into a compact and discriminative representation suitable for temporal modeling. Rather than relying solely on raw keypoint coordinates, additional geometric and relational features are explicitly derived to capture hand configuration, finger articulation, and spatial relationships that are essential for distinguishing visually similar Thai Sign Language (TSL) gestures.

The overall feature extraction pipeline is illustrated in Fig. 3.7, where raw video frames are first processed to obtain hand-centric keypoints, followed by normalization and structured feature construction. The resulting features integrate both spatial configurations and temporal motion characteristics, enabling robust gesture representation while maintaining computational efficiency.

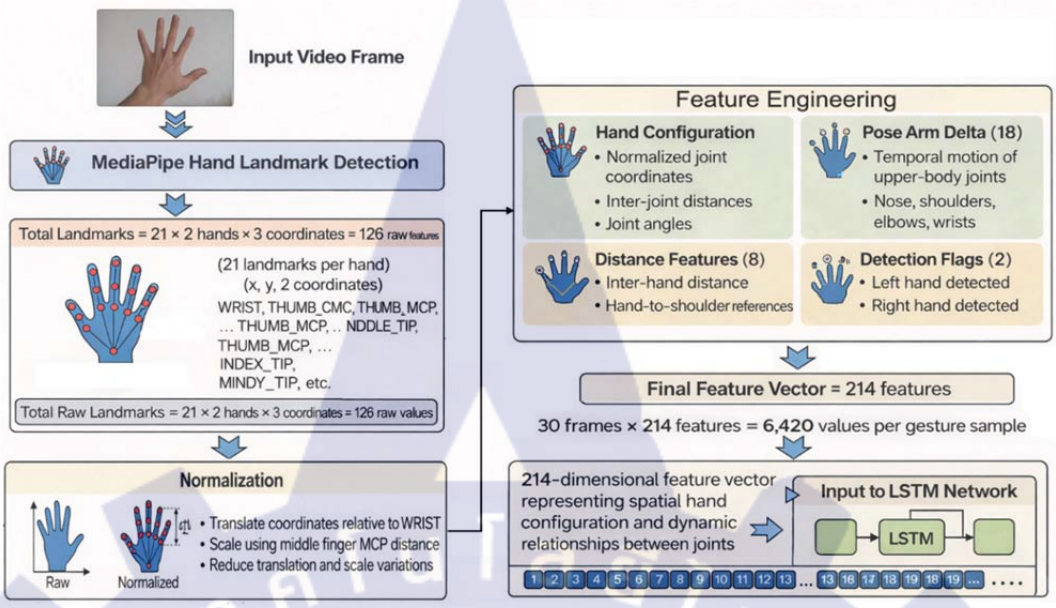


Figure 3.7 Feature engineering pipeline for 214-dimensional keypoint features

The objective of this stage is to enhance recognition robustness while maintaining computational efficiency suitable for real-time operation. These features are designed to capture both intra-hand articulation and inter-joint relationships, enabling the model to distinguish subtle variations in gesture configuration and temporal motion patterns. Feature extraction is performed at the frame level, where each video frame is converted into a fixed-length feature vector. These frame-level features are subsequently assembled into temporal sequences and used as input to the LSTM-based classification model.

Justification of Feature Design

Prior studies have demonstrated that keypoint-based representations significantly improve robustness and generalization in sign language recognition by reducing sensitivity to background variation and illumination changes [21], [35], [43]. While raw keypoint coordinates preserve spatial structure, they are often insufficient for distinguishing gestures with subtle finger articulation differences.

Geometric features such as fingertip distances, finger bending angles, and finger spread measures have been shown to enhance discrimination among visually similar signs by explicitly encoding hand configuration and articulation patterns [30], [31], [39], [40]. Motion-based features derived from temporal changes in keypoints further improve

recognition of dynamic gestures by capturing movement patterns across consecutive frames.

Palm orientation features contribute to capturing global hand posture, which is particularly relevant for gestures involving rotation or directional emphasis [34], [43]. Additionally, thumb-related features are included due to the critical role of thumb positioning in Thai Sign Language, as observed in prior TSL and finger-spelling studies [30], [31].

By integrating normalized keypoint coordinates with geometric, motion-based, and articulation-aware descriptors, the proposed feature set balances representational richness with computational efficiency, making it well suited for real-time sequence modeling using LSTM networks [37], [47].

3.4.1 Feature Engineering Strategy

Thai Sign Language gestures often differ subtly in finger posture, hand openness, thumb orientation, and relative finger movement. To explicitly encode these characteristics, the feature engineering strategy in this study combines three complementary feature groups:

- Normalized skeletal keypoints, which preserve the spatial structure of the hand;
- Geometric descriptors, which capture finger bending, distances, and angular relationships;
- High-level hand configuration features, which describe palm orientation, finger spread, and articulation states.

This multi-level representation enhances discriminative power while maintaining a compact feature dimensionality suitable for real-time deployment.

3.4.2 Base Feature Extraction per Hand

For each detected hand (left and right), a consistent set of features is computed from the normalized MediaPipe landmarks. The same feature definitions are applied symmetrically to both hands to ensure consistency and robustness across one-hand and two-hand gestures.

(1) Normalized Landmark Coordinates

Each hand contributes 21 three-dimensional landmarks, represented by normalized (x, y, z) coordinates, resulting in 63 features per hand. These landmarks preserve the skeletal structure of the hand while remaining invariant to absolute image scale and position.

(2) Fingertip Distance and State Features

Distances between the thumb tip and each fingertip are computed to characterize hand openness and grasping behavior. Binary open/closed flags are derived from these distances to explicitly encode finger extension states, which are particularly important in distinguishing static and semi-dynamic signs.

(3) Finger Geometry and Articulation

For each finger, joint-level geometric features are extracted, including bending angles and corresponding binary indicators that represent whether a finger is flexed or extended. These features help differentiate gestures with similar global shapes but different finger articulations.

(4) Palm Orientation and Finger Spread

The palm normal vector is computed to describe overall hand orientation in three-dimensional space. In addition, finger spread measures are calculated based on angular separation between adjacent fingers, providing cues related to hand openness and expressive posture.

(5) Thumb-Specific Features

Given the importance of thumb positioning in Thai Sign Language, thumb–index angular relationships are explicitly included to enhance discrimination among gestures with similar finger configurations.

The overall feature hierarchy is summarized in Table 3.4.

Table 3.4 Feature Grouping of the 214-Dimensional Input Vector

Group	Feature Group/ Subgroup	Description	No. of Features
A	Base Hand Features	Geometric and articulation features extracted from both hands	186

Table 3.4 Feature Grouping of the 214-Dimensional Input Vector (Continued)

Group	Feature Group/ Subgroup	Description	No. of Features
B	Pose Motion Features (Delta)	Temporal motion features derived from arm pose landmarks	18
C	Hand–Body Spatial Features	Relative hand position to upper-body reference landmarks	8
D	Presence Indicators	Binary indicators describing hand visibility	2
Total Input Features			214

The LSTM model receives a 214-dimensional feature vector for each frame. The 186-dimensional hand configuration feature set, Group A, forms a subset of this vector and is concatenated with Groups B–D to construct the final model input.

Table 3.4 summarizes the hierarchical grouping of all input features used in the proposed recognition model. Among these groups, Group A represents the dominant feature set extracted from hand landmarks. The detailed subgroup composition of Group A is presented in Table 3.5.

Table 3.5 Subgroup Composition of Hand Configuration Features (Group A)

Per-hand Subgroup	Description	No. of Features
A1	Normalized Hand Landmarks (21 points × x, y, z)	63
A2	Fingertip-to-wrist Distances (5 fingertips)	5
A3	Fingertip Extension Flags (5 binary flags)	5

Table 3.5 Subgroup Composition of Hand Configuration Features (Group A) (Continued)

Per-hand Subgroup	Description	No. of Features
A4	Finger Joint Angles (index, middle, ring, pinky)	4
A5	Finger Bent Flags (derived from angles)	4
A6	Finger Lengths Approximation/ tip-to-wrist measures (index, middle, ring, pinky)	4
A7	Palm Orientation Vector (x,y,z)	3
A8	Thumb Extended Flag (angle-based thresholding)	1
A9	Finger Spread Distances (adjacent fingertips: 4 pairs)	4
Total per hand		93

Table 3.5 provides a detailed breakdown of the hand configuration features within Group A. These features describe the spatial, geometric, and articulation properties of hand movements, forming the dominant feature subset of the overall input representation. The mathematical formulations used to derive these engineered features are summarized in Table 3.6.

Table 3.6 List of Engineered Features

Feature Group	Mathematical Representation/ Formulation	Purpose	Rationale for Use
A1) Hand normalization	21 landmarks × (x, y, z)	Normalize hand position and scale relative to the wrist	Reduces variance caused by camera distance and hand size, enabling consistent learning across users

Table 3.6 List of Engineered Features (Continued)

Feature Group	Mathematical Representation/ Formulation	Purpose	Rationale for Use
A2) Fingertip-to-wrist Distances	5 Euclidean distances	Measure finger extension level	Compact representation of hand openness and shape
A3) Finger extension flags	Binary thresholded distances	Binary finger state representation	Converts continuous geometry into interpretable symbolic states
A4) Finger Joint Angle computation	4 fingers × 3 joints	Quantify finger bending at joints	Joint angles provide stable geometric descriptors aligned with phonological properties of sign language
A5) Finger Bending Flags	Binary angle thresholds	Binary bent/straight finger state	Improves robustness against landmark noise
A6) Finger Length Approximation	Tip-to-wrist distance	Approximate relative finger length	Helps disambiguate similar hand shapes
A7) Palm Orientation Vector	3D cross-product vector	Capture palm orientation in 3D	Palm orientation distinguishes visually similar signs
A8) Thumb Extension Flag	Angle at thumb MCP joint	Detect thumb extension state	Thumb posture is highly discriminative for many sign distinctions

Table 3.6 List of Engineered Features (Continued)

Feature Group	Mathematical Representation/ Formulation	Purpose	Rationale for Use
A9) Finger spread distances	Adjacent fingertip distances	Quantify finger separation	Enhances fine-grained handshape discrimination
B) Pose Arm Delta Feature	Temporal difference between normalized arm pose vectors across frames	Capture temporal arm motion dynamics across consecutive frames.	Provides motion context to complement static hand features and improves temporal gesture discrimination
C) Hand-to-Body Distances	Euclidean distance between wrist and body reference landmarks	Encode spatial relationship between the hand and upper-body reference landmarks.	Models phonological location features essential for sign differentiation
D) Hand Presence Flag	Binary indicator of hand detection status	Indicate whether a hand is visible in the current frame.	Enables reliable prediction across varying hand visibility conditions

Table 3.6 summarizes the mathematical concepts used in the feature engineering process.

3.4.3 Mathematical Formulation of Engineered Features

(A1) Hand Landmark Normalization

To achieve scale and translation invariance, each hand landmark $\hat{\mathbf{p}}_{j,t}$ is normalized by translating the wrist joint to the origin and scaling by the Euclidean distance between the wrist and the middle-finger metacarpophalangeal (MCP) joint.

$$\hat{\mathbf{p}}_{j,t} = \frac{\mathbf{p}_{j,t} - \mathbf{w}_t}{\| (\mathbf{m}_t - \mathbf{w}_t)_{(x,y)} \| + \varepsilon} \tag{3.2}$$

where $\mathbf{p}_{j,t}$ denotes the 3D coordinate of the hand landmark with index j ($j = 0, 1, \ldots, 20$) at time step t, $\mathbf{w}_t$ represents the wrist joint coordinate at time t, and $\mathbf{m}_t$ corresponds to the middle-finger metacarpophalangeal (MCP) joint.

The denominator $\| (\mathbf{m}_t - \mathbf{w}_t)_{(x,y)} \|_2$ denotes the Euclidean distance between the wrist and the middle-finger MCP joint projected onto the image plane, which is used as a scale normalization factor. The constant ε is a small positive value added to prevent numerical instability caused by division by zero.

(A2) Tip-to-Wrist Distance

The Euclidean tip-to-wrist distance, denoted as $d_f(t)$, is computed between each fingertip and the wrist to quantify the degree of finger extension at time step t.

$$d_f(t) = || (\mathbf{p}_{\text{tip},f}(t) - \mathbf{p}_{\text{wrist}}(t))_{(x,y)} ||_2 \tag{3.3}$$

where $f \in$ {thumb,index,middle,ring,pinky} denotes the finger index, $\mathbf{p}_{\text{tip},f}(t)$ represents the fingertip landmark coordinate of finger f at time step t, and $\mathbf{p}_{\text{wrist}}(t)$ denotes the wrist landmark coordinate. The Euclidean norm $\|\cdot\|_2$ is computed in the image plane (x, y).

(A3) Finger Extension Flag

A binary finger state is derived by thresholding the fingertip distance.

$$Ext_f(t) = \begin{cases} 1, d_f(t) > \tau_d \\ 0, otherwise \end{cases} \tag{3.4}$$

where $\text{Ext}_f(t) \in \{0,1\}$ denotes the binary extension state of finger f at time step t, $d_f(t)$ represents the fingertip-to-wrist distance defined in Section 3.4.2,

and τ_d is an empirically determined distance threshold used to distinguish between extended and folded finger configurations.

(A4) Finger Joint Angle Computation

Finger bending is modeled using the angle between two adjacent bone vectors at a joint.

$$\theta = \arccos\left(\frac{\overrightarrow{BA} \cdot \overrightarrow{BC}}{\| \overrightarrow{BA} \| \ \| \overrightarrow{BC} \| + \epsilon}\right) \tag{3.5}$$

$$\overrightarrow{BA} = \mathbf{a} - \mathbf{b}, \overrightarrow{BC} = \mathbf{c} - \mathbf{b}$$

where θ denotes the joint bending angle, $\overrightarrow{BA}$ and $\overrightarrow{BC}$ represent the displacement vectors from the joint b to its adjacent joints **a** and **c**, respectively. Here, b corresponds to the joint location at which the angle is computed, while **a** and **c** denote the neighboring joints forming the articulated finger segment. The constant ε is a small positive value introduced to prevent numerical instability due to division by zero.

(A5) Finger Bending Flag

Finger bending is discretized using a joint angle threshold.

$$\text{Bent}_i = \begin{cases} 1, & \theta_i < \tau_\theta \\ 0, & \text{otherwise} \end{cases} \quad i \in \{\text{index}, \ldots, \text{pinky}\} \tag{3.6}$$

where $\text{Bent}_f \in \{0,1\}$ denotes the binary bending state of finger f, θ_f represents the joint angle computed as described in Section 3.4.4, and τ_θ is the joint angle threshold. In this study, $\tau_\theta = 2.2$ radians, and $f \in \{\text{thumb,index,middle,ring,pinky}\}$.

(A6) Finger Length Approximation

Finger length is approximated as a relative geometric descriptor using the Euclidean distance between the fingertip and the wrist, providing a coarse estimate of finger size independent of articulation.

$$L_i(t) = || p_{tip,i}(t) - p_{wrist}(t) ||_2 \quad (3.7)$$

where $L_f(t)$ denotes the approximated length of finger f at time step t, $\mathbf{p}_{\text{tip},f}(t)$ represents the fingertip landmark coordinate of finger f, and $\mathbf{p}_{\text{wrist}}(t)$ denotes the wrist landmark coordinate. The Euclidean norm $\|\cdot\|_2$ is used to compute the spatial distance in three-dimensional space.

(A7) Palm Orientation (Palm Normal Vector)

Palm orientation is encoded using the normalized cross product of two palm vectors to capture the three-dimensional orientation of the palm surface.

$$\mathbf{n}(t) = \frac{\vec{v}_1 \times \vec{v}_2}{\| \vec{v}_1 \times \vec{v}_2 \|_2 + \epsilon} \quad (3.8)$$

$$\vec{v}_1(t) = \mathbf{p}_{index\_MCP}(t) - \mathbf{p}_{wrist}(t)$$

$$\vec{v}_2(t) = \mathbf{p}_{pinky\_MCP}(t) - \mathbf{p}_{wrist}(t)$$

where $\mathbf{n}(t)$ denotes the 3D unit normal vector representing the orientation of the palm at time step t, $\mathbf{v}_1(t)$ and $\mathbf{v}_2(t)$ are displacement vectors originating from the wrist to the metacarpophalangeal (MCP) joints of the index and pinky fingers, respectively. These two vectors span the palm plane, and their cross product yields a vector perpendicular to the palm surface. The normalization term ensures scale invariance, while ε is a small positive constant added to prevent numerical instability when the vectors are nearly collinear.

(A8) Thumb Extension Flag

The thumb extension state is derived from the joint angle at the metacarpophalangeal (MCP) joint, computed as the angle between two vectors originating from the MCP joint.

$$ThumbExtended = \begin{cases} 1, \theta_{thumb(t)} > \tau_{\text{thumb}} \\ 0, otherwise \end{cases} \quad (3.9)$$

where $\theta_{\text{thumb}}(t)$ denotes the angle (in degrees) between the vectors from the thumb MCP joint to the wrist and from the thumb MCP joint to the thumb tip at time step t. The vectors $\mathbf{p}_{\text{wrist}}(t)$, $\mathbf{p}_{\text{thumb\_MCP}}(t)$, and $\mathbf{p}_{\text{thumb\_tip}}(t)$ represent the corresponding 3D landmark coordinates provided by MediaPipe Hands. The binary variable $\text{ThumbExt}(t) \in \{0,1\}$ indicates whether the thumb is extended. In this study, the threshold is set to $\tau_{\text{thumb}} = 150°$.

(A9) Finger Spread Distance

Finger spread is quantified using the Euclidean distance between adjacent fingertip landmarks, capturing the relative separation between neighboring fingers.

$$s_k(t) = || \left(p_{a(k)}(t) - p_{b(k)}(t) \right)_{(x,y)} ||_2 \quad (3.10)$$

where $s_k(t)$ denotes the finger spread distance for the k-th adjacent finger pair at time step t, $\mathbf{p}_{\text{tip},\, a(k)}(t)$ and $\mathbf{p}_{\text{tip},\, b(k)}(t)$ represent the fingertip landmark coordinates of the two neighboring fingers forming pair k. The index pairs $(a(k), b(k))$ correspond to adjacent fingers, specifically $(\text{thumb}, \text{index})$, $(\text{index}, \text{middle})$, $(\text{middle}, \text{ring})$ and $(\text{ring}, \text{pinky})$ The Euclidean norm is computed in the image plane (x, y) to reduce sensitivity to depth noise.

(B) Pose Arm Delta Feature

Temporal arm motion is captured using the frame-to-frame displacement of normalized upper-body pose landmarks.

$$\Delta \mathbf{f}_t = \mathbf{f}_t - \mathbf{f}_{t-1} \quad (3.11)$$

In practice, the pose delta is computed with explicit handling of missing pose landmarks to ensure temporal consistency:

$$\Delta\text{Pose}_t = \begin{cases} 0, \text{if pose landmarks are unavailable at time } t \text{ or } t-1 \\ \text{PoseNorm}_t - \text{PoseNorm}_{(t-1)}, otherwise \end{cases} \quad (3.12)$$

where $\mathbf{f}_t$ denotes a generic feature vector at time step t, PoseNorm_t represents the normalized upper-body pose feature vector, and ΔPose_t denotes the resulting pose arm delta feature. When pose data are missing, the delta is set to zero to avoid introducing spurious motion.

(C) Hand–Body Distance

Spatial relationships between the hands and upper body are captured using wrist-to-body reference distances.

$$d_{hb}(t) = \| \mathbf{p}_{wrist}(t) - \mathbf{p}_{body}(t) \|_2 \quad (3.13)$$

where $d_{hb}(t)$ denotes the Euclidean distance between the wrist and a selected upper-body reference landmark at time step t, such as the nose or shoulder joints, and $\mathbf{p}_{\text{body}}(t)$ represents the corresponding body landmark coordinate. These distances provide spatial context linking hand motion to overall body posture.

(D) Hand Presence Flag

Hand availability is represented using a binary presence indicator to distinguish valid hand detections from missing observations.

$$b_h(t) = \begin{cases} 1, \text{if a hand is detected at time } t \\ 0, otherwise \end{cases} \quad (3.14)$$

where $b_h(t) \in \{0,1\}$ denotes the hand presence flag at time step t. This indicator allows the model to explicitly account for frames in which one or both hands are absent.

To ensure clarity and reproducibility of the proposed feature engineering process, the symbols used in table are defined in Appendix A.

3.4.4 Sequence Assembly and Data Representation

For each gesture clip, the engineered frame-level feature vectors are stacked into a fixed-size tensor of 30 × 214, corresponding to 30 consecutive frames. A class label is associated with each sequence based on its gesture category and stored alongside the feature data.

This structured representation ensures that all gesture samples share a consistent temporal and spatial format, enabling the LSTM model to focus on learning discriminative motion patterns rather than adapting to variable input dimensionality.
The resulting feature sequences form the final input to the temporal classification stage described in the subsequent section.

3.5 LSTM Model Training

After feature construction and engineering, the final stage of the proposed system focuses on training a Long Short-Term Memory (LSTM) network for temporal classification of Thai Sign Language gestures. The LSTM architecture is specifically designed to model sequential dependencies across time while maintaining computational efficiency suitable for real-time deployment.

Each input sample is represented as a fixed-length temporal sequence of 30 frames, where each frame contains 214 engineered features derived from normalized hand landmarks, pose-based motion features, spatial distances, and presence flags, as described in Section 3.4. Consequently, the input to the LSTM network can be expressed as a three-dimensional tensor of shape (30, 214).

3.5.1 Dataset Splitting Strategy:

To ensure unbiased performance evaluation and stable training, the dataset is divided into three mutually exclusive subsets:

• 70% Training set - used to optimize model parameters,

• 15% Validation set - used for hyperparameter tuning and overfitting monitoring,

• 15% Test set - reserved for final performance evaluation.

When sufficient samples per class are available, stratified splitting is applied to preserve class distribution across subsets. This splitting strategy is commonly adopted in sign language recognition research to balance training stability and evaluation reliability [29], [41].

3.5.2 LSTM Network Architecture:

The proposed LSTM model consists of stacked recurrent layers followed by fully connected layers for classification. The overall architecture is illustrated in Figure 3.8.

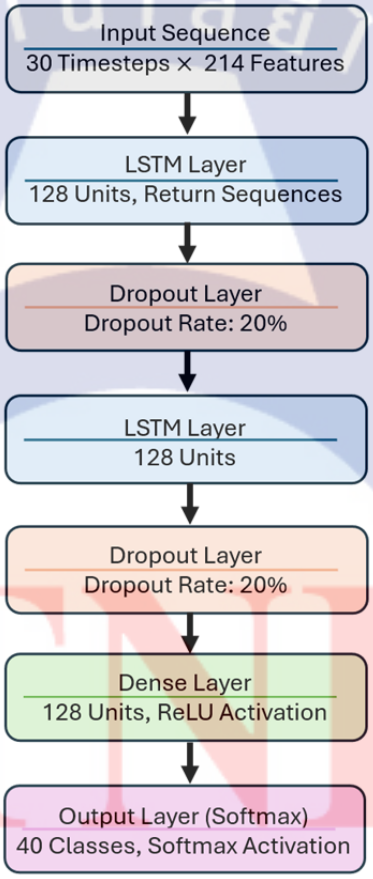


Figure 3.8 Proposed LSTM-based architecture for Thai Sign Language recognition

First LSTM Layer (128 Units, return_sequences = True):

The first recurrent layer contains 128 memory cells and processes the input sequence frame by frame. By enabling return_sequences=True, the layer outputs a full sequence of hidden states across all 30 time steps. This design allows the network to preserve detailed temporal dynamics of hand motion and gesture progression, which is essential for sign language recognition tasks [37], [47].

Dropout Layer (Rate = 0.2):

A dropout layer with a rate of 0.2 follows the first LSTM layer. During training, 20% of neuron activations are randomly deactivated to reduce overfitting and improve generalization. Dropout has been shown to be effective in recurrent neural networks for gesture and sign language recognition [41], [45].

Second LSTM Layer (128 Units):

The second LSTM layer also consists of 128 memory cells, but returns only the final hidden state. This layer aggregates temporal information learned across the entire sequence into a compact representation, effectively summarizing the gesture dynamics into a single feature vector [37], [47].

Second Dropout Layer (Rate = 0.2):

An additional dropout layer with the same rate is applied to further enhance robustness against overfitting, particularly when dealing with limited or imbalanced gesture datasets.

Fully Connected Dense Layer (128 Units, ReLU Activation):

The condensed temporal representation is passed to a dense layer with 128 neurons and a ReLU activation function. This layer transforms the temporal features into a discriminative space suitable for multi-class classification. ReLU activation improves convergence speed and mitigates vanishing gradient issues [37].

Output Layer (Softmax Activation):

The final output layer consists of N neurons, where N corresponds to the number of Thai Sign Language classes defined in the dataset. A Softmax activation

function is used to convert raw outputs into normalized class probabilities that sum to one, enabling probabilistic interpretation of model predictions.

3.5.3 Model Optimization and Training Configuration:

The model is trained using the Adam optimizer with a learning rate of 3×10^{-4}, which provides a balance between fast convergence and training stability. Gradient clipping is applied to prevent exploding gradients, which can occur in deep recurrent networks.

The categorical cross-entropy loss function is employed to support multi-class classification, and training is conducted for 50 epochs with a batch size of 32. Model performance is monitored using validation accuracy and loss curves, which are later analyzed to detect overfitting or underfitting behavior.

Architectural Rationale

The stacked LSTM architecture is designed to support hierarchical temporal modeling, where lower recurrent layers focus on capturing short-term motion patterns, while higher layers learn longer-range temporal dependencies across gesture sequences. This hierarchical representation is particularly suitable for sign language recognition, where both fine-grained finger movements and overall gesture progression play critical roles.

When combined with the engineered spatial and motion-based features described in Section 3.4, the proposed architecture achieves a balance between representational expressiveness, robustness to variation, and computational efficiency. Similar multi-layer LSTM architectures have been reported to perform effectively in both isolated and continuous sign language recognition tasks, especially when applied to skeletal keypoint-based representations [29], [37], [41], [43], [47].

3.6 Model Evaluation

To objectively assess the performance of the proposed Thai Sign Language recognition system, a set of standard classification evaluation metrics is employed. These metrics are selected to reflect not only overall prediction accuracy but also class-level

reliability and robustness, which are particularly important in sign language recognition tasks where gesture classes may exhibit visual similarity and data imbalance.

3.6.1 Accuracy

Accuracy measures the proportion of correctly classified samples among all evaluated samples. It provides a general indication of the model's overall prediction performance and is defined as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{3.15}$$

While accuracy is intuitive and widely used, it may not fully reflect model performance when class distributions are imbalanced or when certain gesture classes are more difficult to recognize than others. Therefore, additional metrics are required to provide a more comprehensive evaluation [2].

3.6.2 Precision, Recall, and F1-Score

To better evaluate class-level performance, precision, recall, and F1-score are used. These metrics are particularly important for sign language recognition systems, where misclassification of visually similar gestures can significantly impact usability [41], [45], [50].

Recall measures how effectively a machine learning model identifies true positives out of all actual positive samples in the dataset. In other words, it reflects the model's ability to correctly recognize the ground truth instances belonging to a particular class. A high recall indicates that the system successfully captures most of the positive labels, aligning with the goal of minimizing missed detections and is defined as:

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{3.16}$$

Where:

• TP (True Positive) represents the number of correctly predicted instances for a given class.

• FN (False Negative) represents the number of instances belonging to the target class that are incorrectly predicted as another class.

A high recall indicates that the system successfully detects most instances of a gesture class, which is critical in assistive communication scenarios where missed detections can disrupt interpretation.

Precision reflects how many of the predicted positive instances are actually correct, measuring the reliability of the model's predictions.

The F1-score combines precision and recall into a single metric using their harmonic mean, providing a balanced measure that accounts for both false positives and false negatives. While accuracy offers a general overview of overall model performance, the F1-score is more reliable when evaluating complex or imbalanced datasets, as it reflects both the correctness and completeness of predictions:

$$\text{F1-score} = 2 \times \frac{(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \tag{3.17}$$

Compared to accuracy alone, the F1-score offers a more reliable indicator of performance when dealing with complex gesture sets and uneven class distributions, as commonly observed in sign language datasets [41], [45], [50].

3.6.3 Confusion Matrix

In addition to scalar performance metrics, a confusion matrix is employed to provide a detailed analysis of classification results across all gesture classes. The confusion matrix illustrates the number of correct predictions along the diagonal, as well as misclassifications between specific gesture pairs.

This analysis is particularly useful for identifying systematic errors, such as confusion between gestures with similar hand shapes, orientations, or motion patterns. Insights obtained from the confusion matrix help reveal limitations of the feature

representation and guide future improvements in feature engineering and model design [41], [49].

3.6.4 Evaluation Strategy

In this study, accuracy, precision, recall, F1-score, and confusion matrix analysis are used jointly to evaluate the proposed system. This combined evaluation strategy ensures that the model demonstrates not only high overall accuracy but also consistent and fair performance across different Thai Sign Language gesture classes.

Such a multi-metric evaluation approach is widely adopted in recent sign language recognition research and is essential for assessing real-world applicability and robustness of real-time recognition systems [49], [50].

3.7 Output Prediction

This section describes how the trained LSTM model produces class-level predictions from input feature sequences and how these predictions are interpreted during both offline testing and real-time operation [50], [51], [52].

For each input sequence of 30 frames, the model outputs a probability distribution over all predefined Thai Sign Language classes using a Softmax activation function. The predicted class corresponds to the class with the highest probability score.

During offline evaluation, predicted labels are compared with ground truth annotations from the test dataset to compute performance metrics and confusion matrices, which are demonstrated in Chapter 4. In real-time operation, predictions are generated continuously as new frames are processed, enabling responsive gesture recognition suitable for assistive communication scenarios.

Chapter 4
Experimental Setup and Results

This chapter presents the experimental setup and evaluation results of the proposed real-time Thai Sign Language recognition system based on skeletal keypoints and Long Short-Term Memory (LSTM) networks. The experiments were designed to assess both recognition performance and real-time feasibility under practical usage conditions. Using a dataset of Thai Sign Language gestures, the system was trained and evaluated by analyzing classification accuracy, F1-score, and overall behavioral consistency during live webcam testing. The results obtained from the experiments are further analyzed and compared with findings from related Thai Sign Language studies in order to demonstrate the effectiveness and limitations of the proposed approach.

4.1 Experimental Environment

The experiments were conducted in a controlled computing environment designed to reflect realistic deployment conditions for a real-time Thai Sign Language recognition system. Rather than relying on high-end computing resources, the experimental setup focused on consumer-grade hardware to evaluate whether the proposed approach can operate effectively in practical, real-world scenarios, such as assistive communication applications.

The system was implemented using Python, with MediaPipe employed for real-time skeletal keypoint extraction from webcam video input, and TensorFlow (Keras) used for training and evaluating the LSTM-based classification model. Data preprocessing, feature extraction, and model experimentation were carried out using standard scientific computing libraries, including NumPy and Pandas. Development and experimentation were performed through Jupyter Notebook, which enabled interactive analysis and rapid iteration during model development.

Importantly, GPU acceleration was not required in this study. MediaPipe's hand landmark detection operates efficiently on CPU, and the LSTM model was designed to be lightweight, using engineered skeletal features rather than high-dimensional raw image

data. This design choice allows the system to achieve real-time performance without specialized hardware, supporting the goal of accessibility and ease of deployment.

The detailed experimental environment is summarized in Table 4.1

Table 4.1 Experimental Environment

Component	Specification
Execution Platform	Local workstation
Operating System	Windows 11 (64-bit)
CPU	Intel Core i5 / AMD Ryzen equivalent
RAM	16 GB
Camera	Standard USB webcam (30 FPS)
Programming Language	Python
Frameworks & Libraries	MediaPipe, TensorFlow (Keras), OpenCV
Data Processing Tools	NumPy, Pandas
Development Tool	Jupyter Notebook

4.2 Experimental Results

This section presents the experimental results of the proposed Thai Sign Language recognition system. The objective of these experiments is to investigate the effect of different LSTM architectural configurations and learning-rate settings on recognition performance when using skeletal keypoint-based temporal features.

All models were trained using the same dataset, feature representation, and fixed data partition to ensure a fair comparison. Model performance was evaluated on a held-out test set that was not used during training. Standard classification metrics including accuracy, precision, recall, and F1-score were employed to quantify recognition performance.

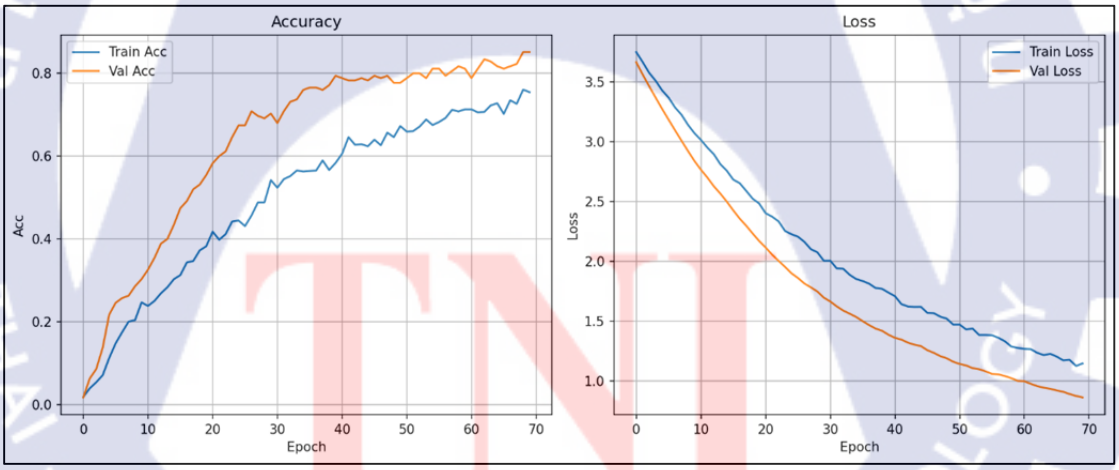
A series of fine-tuning experiments were conducted using three LSTM configurations: (1) a single-layer LSTM with 64 units, (2) a two-layer LSTM with 128 and 64 units, and (3) a two-layer LSTM with 128 units in both layers. For each architecture, two learning-rate configurations (Adam optimizer with learning rates of 1×10^{-4} and 3×10^{-4}) were evaluated. Training and validation accuracy and loss curves are used to

analyze convergence behavior and generalization trends. The results are presented in the following subsections.

4.2.1 Performance Evaluation of a Single-Layer LSTM (64 Units)

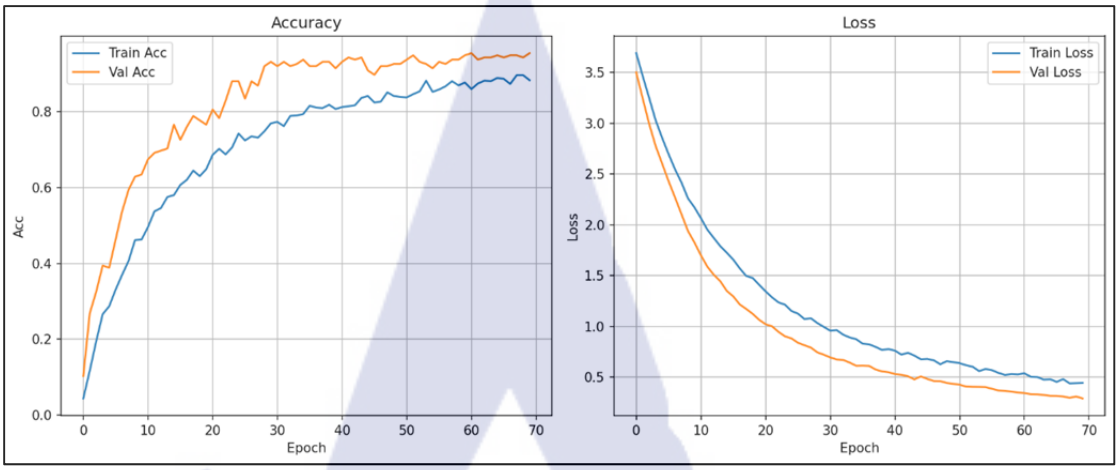
This experiment evaluates a baseline temporal modeling architecture consisting of a single-layer Long Short-Term Memory (LSTM) network with 64 hidden units. The objective of this configuration is to assess the recognition performance of a lightweight model suitable for real-time Thai Sign Language recognition, particularly in scenarios with limited computational resources.

The model was trained using the Adam optimizer under two learning-rate configurations: 3×10^{-4} and 1×10^{-4}. For both settings, training and validation accuracy and loss curves indicate stable convergence behavior without signs of severe overfitting, as illustrated in Figure 4.1. Specifically, Figure 4.1(a) illustrates the training and validation accuracy and loss obtained using a learning rate of 1×10^{-4}, while Figure 4.1(b) shows the corresponding results for a learning rate of 3×10^{-4}.



(a)

Figure 4.1 Training and validation performance of the single-layer LSTM (64 Units) under the two learning-rate configurations: (a) learning rate = 1×10^{-4}, (b) learning rate = 3×10^{-4}



(b)

Figure 4.1 Training and validation performance of the single-layer LSTM (64 Units) under the two learning-rate configurations: (a) learning rate = 1×10^{-4}, (b) learning rate = 3×10^{-4} (Continued)

As observed in Figure 4.1(a), the model trained with a learning rate of 1×10^{-4} demonstrates a gradual and stable convergence pattern. The validation accuracy increases smoothly across epochs, accompanied by a consistent reduction in validation loss, indicating stable learning behavior and reduced sensitivity to parameter updates.

In contrast, Figure 4.1(b) shows that the configuration using a higher learning rate of 3×10^{-4} achieves faster initial convergence. However, slight fluctuations in the validation curves can be observed during later training stages, suggesting marginal instability as the optimization approaches convergence.

Although the single-layer LSTM achieves reasonable recognition accuracy under both learning-rate settings, its overall performance remains constrained by limited temporal modeling capacity. This limitation is particularly evident for gestures involving complex motion dynamics or extended temporal dependencies. The quantitative classification results for this architecture are summarized in Table 4.2, which provides a direct comparison of performance metrics across learning-rate configurations.

Table 4.2 Classification Performance Comparison of a Single-Layer LSTM (64 Units) under Different Learning Rates

Learning Rate	Accuracy	Precision	Recall	F1-score
1×10^{-4}	0.903	0.909	0.904	0.906
3×10^{-4}	0.938	0.952	0.942	0.947

Table 4.2 summarizes the classification performance of the single-layer LSTM model with 64 hidden units under two learning-rate configurations using the Adam optimizer. The results are reported on a held-out test dataset and include accuracy, precision, recall, and F1-score.

While the single-layer LSTM demonstrates stable learning behavior and provides a competitive baseline for Thai Sign Language recognition, its representational capacity is inherently constrained by the shallow network depth. Consequently, the model shows limited effectiveness in capturing gestures that involve complex temporal dependencies or subtle motion transitions. This observation motivates the investigation of deeper LSTM architectures, which are examined in the following subsections.

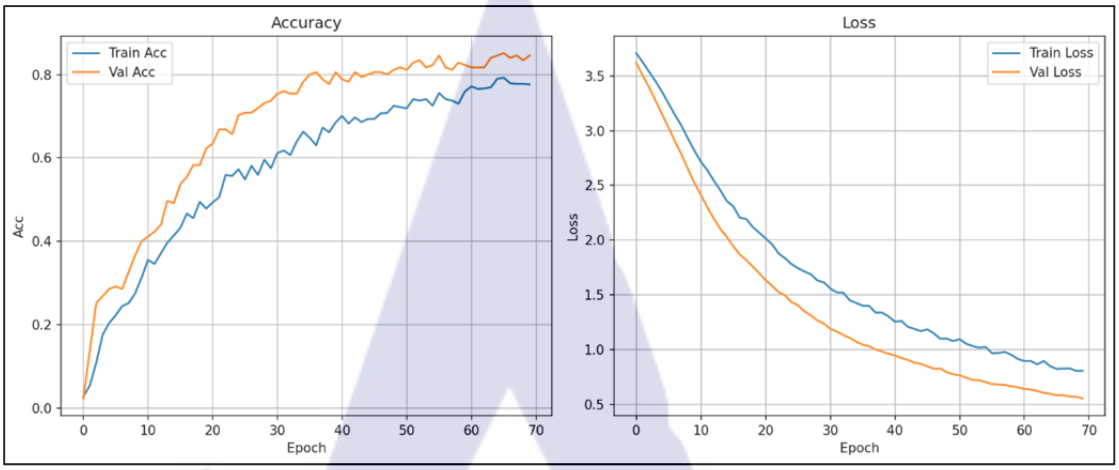
4.2.2 Performance Evaluation of a Two-Layer LSTM (128–64 Units)

This experiment investigates a deeper LSTM architecture composed of two recurrent layers with 128 hidden units in the first layer and 64 hidden units in the second layer. The primary objective of this configuration is to enhance temporal feature abstraction by introducing hierarchical temporal modeling, while maintaining a controlled level of model complexity suitable for real-time applications.

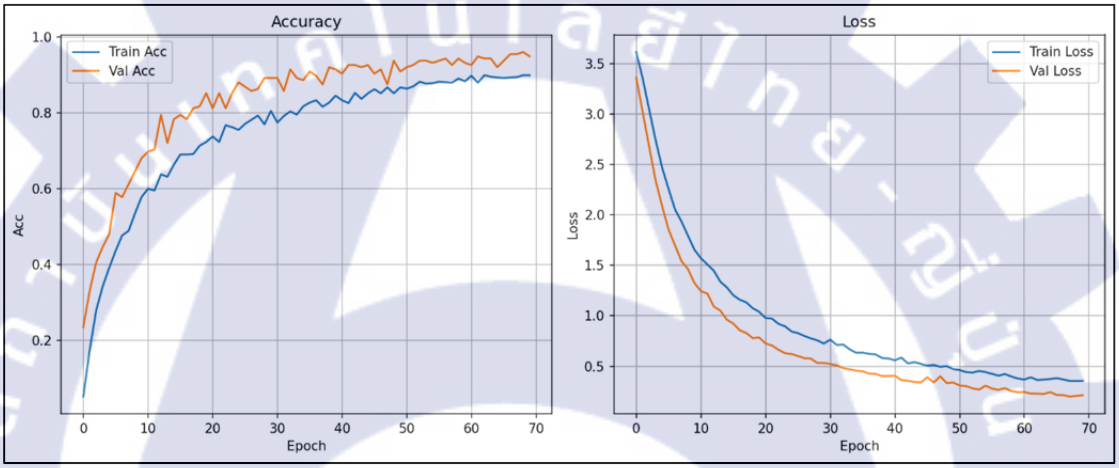
Consistent with the baseline experiment, the Adam optimizer was employed with two learning-rate configurations, namely 1×10^{-4} and 3×10^{-4}, to examine their effects on convergence behavior and classification performance.

Figure 4.2 presents the training and validation learning curves of the two-layer LSTM (128–64 units) under the two learning-rate settings.

Specifically, Figure 4.2(a) illustrates the accuracy and loss curves obtained using a learning rate of 1×10^{-4}, while Figure 4.2(b) shows the corresponding results for a learning rate of 3×10^{-4}.



(a)



(b)

Figure 4.2 Training and validation performance of a two-layer LSTM (128-64 units) under different learning-rate configurations: (a) learning rate = 1×10^{-4}, (b) learning rate = 3×10^{-4}

As observed in Figure 4.2, the two-layer LSTM demonstrates faster convergence and improved validation performance compared to the single-layer baseline. The configuration using a learning rate of 1×10^{-4} exhibits smoother convergence and more stable validation loss, whereas the higher learning rate of 3×10^{-4} achieves quicker initial learning but shows slightly increased fluctuation during later training epochs.

The quantitative classification performance of this architecture is summarized in Table 4.3. Compared to the single-layer LSTM, the two-layer configuration achieves consistently higher recognition performance, particularly in terms of recall and F1-score, indicating improved sensitivity to gesture variations and temporal dynamics.

Table 4.3 Classification performance comparison of a two-layer LSTM (128-64 units) under Different Learning Rates

Learning Rate	**Accuracy**	**Precision**	**Recall**	**F1-score**
1×10^{-4}	0.943	0.935	0.945	0.940
3×10^{-4}	0.972	0.975	0.974	0.974

Overall, these results demonstrate that introducing an additional LSTM layer enhances the model's ability to capture complex temporal dependencies in Thai Sign Language gestures. While minor sensitivity to learning-rate selection is observed, the two-layer architecture consistently outperforms the single-layer baseline. This observation motivates further investigation of deeper LSTM configurations, as discussed in the following subsection.

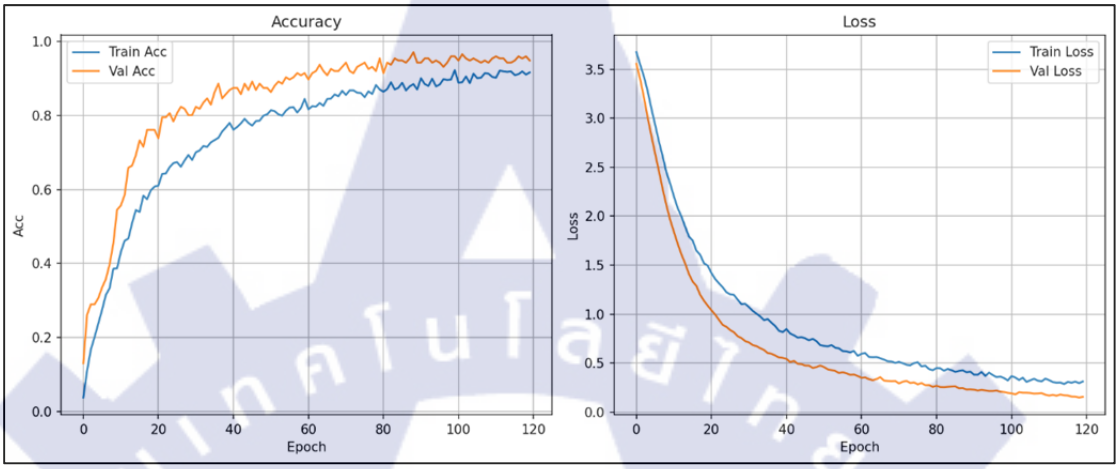
4.2.3 Performance Evaluation of a Two-Layer LSTM (128–128 Units)

This experiment evaluates the deepest LSTM architecture considered in this study, consisting of two recurrent layers with 128 hidden units in both the first and second layers. The objective of this configuration is to further enhance temporal representation capacity by increasing model depth, enabling more comprehensive modeling of complex and long-range temporal dependencies in Thai Sign Language gestures.

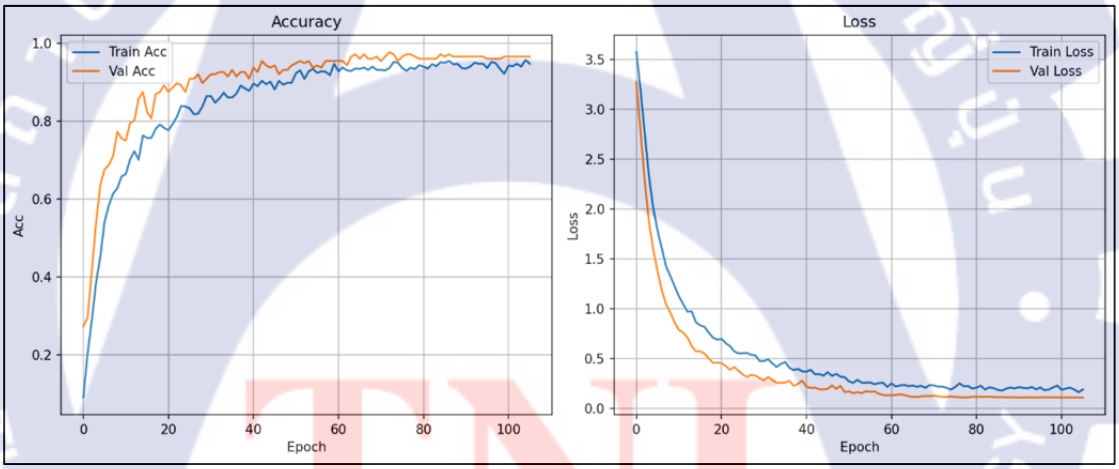
As in the previous experiments, the Adam optimizer was applied using two learning-rate settings, 1×10^{-4} and 3×10^{-4}, to analyze their effects on convergence behavior and classification performance under a higher-capacity network.

Figure 4.3 presents the training and validation learning curves of the two-layer LSTM (128–128 units) under the two learning-rate configurations.

Specifically, Figure 4.3(a) shows the accuracy and loss curves obtained using a learning rate of 1×10^{-4}, while Figure 4.3(b) illustrates the corresponding results for a learning rate of 3×10^{-4}.



(a)



(b)

Figure 4.3 Training and validation performance of a two-layer LSTM (128-128 units) under different learning-rate configurations: (a) learning rate = 1×10^{-4}, (b) learning rate = 3×10^{-4}

The learning curves indicate that the two-layer LSTM with 128 units in both layers achieves stable convergence under both learning-rate settings. Compared to the shallower architectures examined earlier, this configuration demonstrates higher

validation accuracy and lower validation loss, reflecting improved modeling of temporal gesture dynamics. The model trained with a learning rate of 1×10^{-4} exhibits smoother convergence and reduced fluctuation during later training stages, whereas the configuration using 3×10^{-4} converges more rapidly but shows slight variability near convergence.

The quantitative classification performance of this architecture is summarized in Table 4.4. Among all evaluated configurations, the two-layer LSTM (128–128 units) achieves the highest overall recognition performance across accuracy, precision, recall, and F1-score metrics. These results suggest that the increased representational capacity provided by the deeper architecture enables more effective discrimination of gestures with subtle temporal variations.

Table 4.4 Classification performance comparison of a two-layer LSTM (128-128 units) under Different Learning Rates

Learning Rate	**Accuracy**	**Precision**	**Recall**	**F1-score**
1×10^{-4}	0.955	0.965	0.955	0.960
3×10^{-4}	0.977	0.984	0.977	0.980

Overall, the experimental results demonstrate that increasing LSTM depth and hidden-unit capacity leads to consistent performance improvements for Thai Sign Language recognition. The two-layer LSTM with 128 hidden units in both layers provides the most favorable balance between temporal modeling capability and training stability, and is therefore selected as the primary model for subsequent performance analysis.

4.3 Performance Analysis

This section presents a detailed analysis of the proposed Thai Sign Language (TSL) recognition system based on the best-performing LSTM configuration identified in the experimental results. The evaluation focuses on classification behavior, error characteristics, and class-wise performance, together with practical considerations for real-time deployment.

4.3.1 Confusion Matrix Analysis

The confusion matrix of the proposed model is illustrated in Fig. 4.4, providing a comprehensive view of classification outcomes across all Thai Sign Language gesture classes.

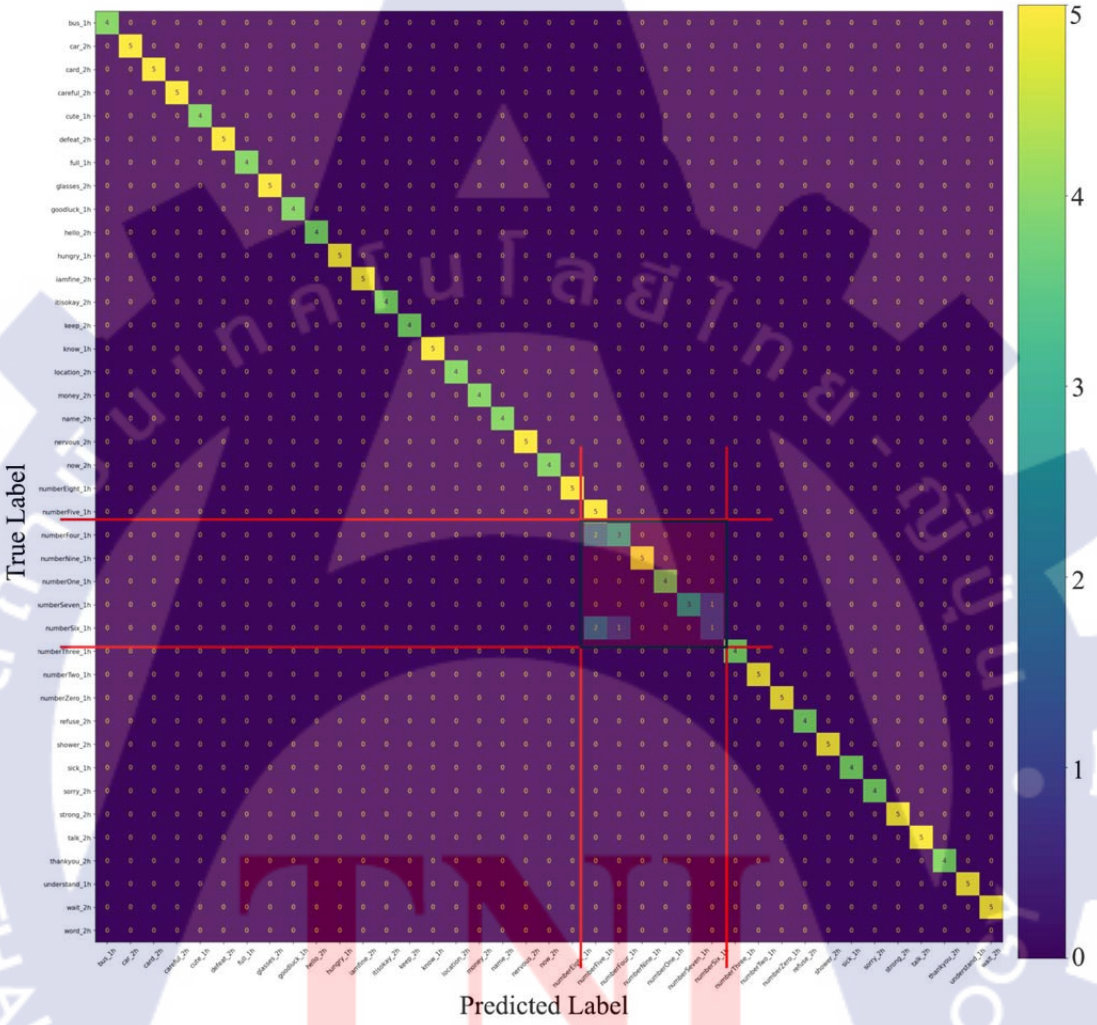


Figure 4.4 Confusion matrix of the proposed Thai Sign Language recognition system using the best-performing two-layer LSTM (128-128) model

As shown in Fig. 4.4, the matrix exhibits strong diagonal dominance, indicating that the majority of gesture samples are correctly classified. This pattern suggests that the LSTM-based temporal model, combined with MediaPipe keypoint-

based features, effectively captures both spatial hand configurations and temporal motion dynamics.

Gesture classes with distinctive hand shapes or clear motion patterns achieve consistently high recognition accuracy, with minimal confusion observed across categories. This indicates that the engineered keypoint features provide sufficient discriminative information for reliable gesture recognition.

However, a limited number of off-diagonal entries are present, reflecting misclassification among visually similar gestures. In particular, numerical gestures such as six and seven show noticeable confusion. These errors are likely caused by subtle differences in finger articulation and slight variations in motion patterns, which are inherently difficult to distinguish using keypoint representations alone.

Such behavior is consistent with prior findings in sign language recognition, where fine-grained gesture variations and limited training samples per class contribute to classification ambiguity [27]. Despite these localized errors, the overall structure of the confusion matrix confirms that the proposed system maintains a high level of classification reliability.

4.3.2 Class-wise Performance Analysis

To further evaluate recognition performance, class-wise precision, recall, and F1-score are summarized in Table 4.4.

Under the best-performing configuration, the model achieves an overall accuracy of 0.977 and an F1-score of 0.980, indicating strong recognition capability across the dataset. The close alignment between accuracy and F1-score suggests that the model does not suffer significantly from class imbalance, and maintains consistent performance across different gesture categories. These results suggest that the model generalizes effectively across diverse Thai Sign Language gestures.

Most gesture classes achieve consistently high precision and recall values, reflecting balanced classification performance. High precision indicates that false-positive predictions are minimal, while strong recall demonstrates the model's ability to correctly identify gesture instances.

Nevertheless, certain classes exhibit comparatively lower performance. In particular, numerical gestures such as six show reduced recall and F1-score. These cases typically involve subtle finger configurations and overlapping motion patterns, which are more challenging to distinguish using keypoint-based features.

Despite these challenges, the overall class-wise metrics remain stable and well-balanced. The close alignment between accuracy and F1-score further indicates that the model does not exhibit bias toward specific gesture categories, supporting its reliability for practical applications.

4.3.3 Implications for Real-Time Recognition

From a practical perspective, the proposed system demonstrates strong potential for real-time deployment. The model achieves stable classification performance while maintaining low computational overhead due to the use of keypoint-based representations instead of raw image inputs.

Table 4.5 Real-Time Performance Metrics of the Proposed Thai Sign Language Recognition System

Metric	Value
Average Pipeline FPS	7.31 frames/s
Average Inference Latency	72 ms
Average Pipeline Latency	127 ms
Average System CPU Usage	22%
Hardware Platform	Intel-based laptop CPU

As summarized in Table 4.5, the system operates at an average processing speed of 7.31 frames per second, with an inference latency of 72 ms and an overall pipeline latency of 127 ms. These results indicate that the system is capable of near real-time operation on standard CPU-based hardware without requiring GPU acceleration.

The relatively low CPU utilization further highlights the efficiency of the proposed approach, making it suitable for deployment on resource-constrained devices such as low-specification laptops or embedded platforms.

While most misclassifications are limited to a small subset of visually similar gestures, they do not significantly affect overall usability in real-time scenarios. This suggests that the system can provide reliable feedback for assistive communication applications.

However, real-time performance remains influenced by environmental factors such as lighting conditions, hand distance from the camera, and variations in gesture execution speed. These factors may affect keypoint detection stability and, consequently, prediction accuracy. In addition, the observed pipeline latency reflects a trade-off between model complexity and responsiveness, indicating potential areas for further optimization.

Overall, the results confirm that the proposed system achieves a practical balance between recognition accuracy and computational efficiency, supporting its applicability in real-world Thai Sign Language recognition scenarios.

4.3.4 Discussion

The experimental results demonstrate that the proposed Thai Sign Language (TSL) recognition system achieves strong overall performance while maintaining computational efficiency suitable for real-time deployment. The combination of Media Pipe keypoint-based representation and LSTM-based temporal modeling provides an effective balance between recognition accuracy and system complexity.

Compared with conventional image-based approaches, the use of keypoint representations significantly reduces input dimensionality and minimizes sensitivity to background variations and illumination changes. This allows the model to focus on essential gesture characteristics, including hand configuration and temporal motion patterns, rather than irrelevant visual information. As a result, the system achieves high recognition performance while remaining lightweight and suitable for CPU-based execution.

The effectiveness of temporal modeling is reflected in the improved performance of deeper LSTM architectures. The two-layer LSTM configuration demonstrates a stronger ability to capture sequential dependencies compared with the single-layer baseline, particularly for gestures involving dynamic transitions. This observation highlights

the importance of temporal context in sign language recognition, where subtle motion patterns often carry critical semantic meaning.

Despite these strengths, several limitations are observed. Misclassification primarily occurs among gesture classes with similar hand configurations, especially numerical signs such as six and seven. These errors indicate that fine-grained finger articulation remains challenging to distinguish using keypoint-based features alone, particularly when differences are subtle and temporally overlapping.

In addition, the dataset used in this study is relatively limited in terms of signer diversity and environmental variability. Although controlled variations are introduced during data collection, the model may experience reduced generalization when applied to unseen users or more complex real-world conditions. This limitation is consistent with prior studies, where models trained on constrained datasets often exhibit performance degradation in practical deployment scenarios.

From a system perspective, the results highlight an inherent trade-off between model complexity, recognition accuracy, and real-time responsiveness. While more advanced architectures, such as transformer-based or graph-based models, may achieve higher recognition accuracy, they typically require substantially greater computational resources. In contrast, the proposed LSTM-based approach achieves competitive performance while maintaining real-time capability on standard hardware.

Overall, the findings suggest that keypoint-based temporal modeling is a practical and effective approach for real-time Thai Sign Language recognition. Future improvements may focus on expanding the dataset, increasing signer diversity, and exploring hybrid feature representations or advanced temporal models to further enhance discrimination of visually similar gestures while preserving computational efficiency.

4.3.5 Comparison with Related Work

To further evaluate the effectiveness of the proposed approach, a comparison with representative studies in sign language recognition is presented in Table 4.6. Due to differences in datasets, gesture vocabularies, and experimental protocols, direct performance comparison should be interpreted with caution. Nevertheless, the comparison provides useful insights into the relative strengths of the proposed method.

Table 4.6 Comparison with Related Work

Study	Method	Dataset	Accuracy
CNN-based TSL [5]	CNN	Finger spelling	0.813
MediaPipe + BiLSTM [14]	Landmark + BiLSTM	TSL gestures	0.910
CNN-LSTM SLRNet [51]	CNN + LSTM	Sign language dataset	0.863
Proposed Method	**MediaPipe + engineered features + LSTM**	**TSL dataset (40 gestures)**	**0.977**

As shown in Table 4.6, the proposed system achieves an overall accuracy of 0.977 on the Thai Sign Language dataset, demonstrating competitive performance compared to existing approaches. In contrast to conventional CNN-based methods, which rely on raw image inputs, the proposed approach utilizes MediaPipe keypoint-based representations to significantly reduce input dimensionality and improve robustness to background variations.

Compared with hybrid CNN–LSTM architectures, the proposed method achieves comparable or higher recognition performance while requiring substantially lower computational resources. This highlights the advantage of combining engineered keypoint features with LSTM-based temporal modeling for efficient gesture recognition.

In addition, several prior studies focus on limited gesture sets, such as finger spelling or numerical signs. In contrast, the proposed system supports a broader set of Thai Sign Language gestures, enhancing its applicability in real-world communication scenarios.

From a practical perspective, the proposed approach offers a favorable balance between recognition accuracy and computational efficiency. While more complex architectures may achieve marginally higher accuracy under controlled conditions, they often require GPU acceleration and higher computational cost. The proposed system, however, maintains strong performance while remaining suitable for real-time deployment on standard CPU-based hardware.

Overall, the comparison suggests that the proposed method provides a practical and scalable solution for Thai Sign Language recognition, particularly in scenarios where computational resources are limited and real-time performance is required.

Chapter 5
Conclusion and Future Works

5.1 Key Findings

This study presented a real-time Thai Sign Language recognition system based on MediaPipe keypoint representation and LSTM-based temporal modeling. By leveraging structured keypoint features instead of raw image data, the proposed approach effectively captures both spatial hand configurations and temporal motion dynamics while maintaining low computational complexity.

Experimental results demonstrate that the system achieves reliable recognition performance across a diverse set of Thai Sign Language gestures. The model maintains strong classification accuracy and balanced performance across gesture classes, while also achieving near real-time operation on standard CPU-based hardware.

The findings highlight that keypoint-based feature engineering, combined with lightweight temporal modeling, provides an effective solution for practical sign language recognition systems. The proposed framework is particularly suitable for deployment in accessibility-oriented applications, where computational resources are limited.

Despite these promising results, challenges remain in distinguishing visually similar gestures and improving generalization under diverse real-world conditions. Future work may focus on expanding the dataset, increasing signer diversity, and exploring advanced modeling techniques to further enhance recognition performance.

Overall, this research provides a practical and scalable foundation for real-time Thai Sign Language recognition and contributes to the development of assistive communication technologies for the Thai Deaf community. This work demonstrates that efficient, real-time sign language recognition can be achieved through carefully designed keypoint-based representations, paving the way for accessible and deployable solutions in real-world communication environments.

5.2 Contributions and Future Works

5.2.1 Research Contributions

This research makes several important contributions to the field of Thai Sign Language recognition and real-time gesture-based communication systems.

First, this study proposes a real-time Thai Sign Language recognition framework based on skeletal keypoints and Long Short-Term Memory (LSTM) networks. By focusing on hand landmark representations rather than raw image data, the proposed system achieves effective gesture recognition while maintaining low computational complexity. This design enables real-time performance on consumer-grade hardware without GPU acceleration, supporting accessible and practical deployment.

Second, the research demonstrates that temporal modeling of skeletal keypoints is well suited for capturing both static hand configurations and dynamic motion patterns inherent in Thai Sign Language gestures [3], [14], [27]. The experimental results confirm that sequence-based learning can reliably distinguish between multiple gesture classes, even when subtle temporal variations are present.

Third, this study provides a systematic and transparent evaluation methodology using standard classification metrics and confusion matrix analysis. By employing macro-averaged precision, recall, and F1-score, the evaluation ensures balanced assessment across all gesture classes, including those with limited training samples. This contributes to reproducibility and methodological clarity for future Thai Sign Language research.

Finally, the proposed system establishes a practical foundation for assistive communication applications. Its lightweight design, real-time capability, and robustness to common environmental variations make it suitable for real-world usage scenarios, such as educational tools and accessibility support systems for individuals with hearing impairments.

5.2.2 Future Works

Although the proposed system demonstrates promising performance, several directions remain for future research and development.

One important direction involves dataset expansion. Increasing the number of gesture classes, recording sessions, and signers would improve model generalization and reduce misclassification between visually similar gestures. Incorporating greater variation in hand size, signing speed, and environmental conditions would further enhance robustness.

Another potential extension is signer-independent evaluation. Future studies could explicitly separate training and testing data by individual signers to assess generalization performance on unseen users, providing stronger evidence of real-world applicability.

In terms of modeling techniques, future work may explore more advanced temporal architectures, such as Bidirectional LSTM, Temporal Convolutional Networks, or lightweight Transformer-based models. These approaches may further improve recognition accuracy, particularly for gestures with complex temporal dependencies.

Additionally, extending the system from isolated gesture recognition to continuous Thai Sign Language recognition represents an important research direction. This would require gesture segmentation, temporal alignment, and sentence-level interpretation, which are essential components of full sign language translation systems.

Finally, future work could focus on deployment and optimization for embedded or mobile platforms, such as Raspberry Pi or smartphones. Applying model compression and optimization techniques could improve portability and energy efficiency, enabling broader adoption in real-world assistive applications.

References

References

[1] Department of Empowerment of Persons with Disabilities, *"Statistics of Persons with Disabilities in Thailand,"* [Online]. Available: https://dep.go.th. [Accessed : August 2, 2025].

[2] B. Alsharif et al., "Real-time american sign language interpretation using deep learning and keypoint tracking," *Sensors,* vol. 25, no. 7, pp. 2,138-2,159, March 2025.

[3] S. Kamble, *"SLRNet : A Real-Time LSTM-Based Sign Language Recognition System,"* [Online]. Available: https://www.researchgate. net/publication/392716 727. [Accessed : September 2, 2025].

[4] A. Krishna et al., "CNN-based real-time ASL gesture recognition and communication aid," *International Conference on Computing Technologies,* ICOCT 2025, Bengaluru, India, June 13-14, 2025, pp. 1-6.

[5] W. Vijitkunsawat et al., "Video-based sign language digit recognition for the thai language : A new dataset and method comparisons," *Proceedings of the International Conference on Pattern Recognition Applications and Methods,* ICPRAM 2023, Lisbon, Portugal, February 22-24, 2023, pp. 775-782.

[6] C. Damrongekarun et al., "Development of thai sign language detection and conversion system into thai with deep learning," *KKU Science Journal,* vol. 51, no. 3, pp. 216-225, September 2023.

[7] T. Pariwat and P. Seresangtakul, "Multi-stroke thai finger-spelling sign language recognition system with deep learning," *Symmetry,* vol. 13, no. 2, p. 262-281, February 2021.

[8] W. Vijitkunsawat and T. Racharak, "A comprehensive benchmark and evaluation of thai finger spelling in multi-modal deep learning models," *IEEE Symposium on Computers & Informatics,* ISCI 2024, Kuala Lumpur, Malaysia, October 12, 2024, pp. 158,079-158,093.

[9] S. Renjith et al., "An effective skeleton-based approach for multilingual sign language recognition," *Engineering Applications of Artificial Intelligence,* vol. 143, no. 1, pp. 1-12, March 2025.

[10] Google, *"MediaPipe Solutions Guide,"* [Online]. Available: https://ai.google.dev/edge/mediapipe/solutions/guide. [Accessed : September 22, 2025].

[11] W. Jintanachaiwat et al., "Using LSTM to translate thai sign language to text in real time," *Discover Artificial Intelligence,* vol. 4, no. 1, pp. 17-28, February 2024.

[12] G. M. Rao et al., "Sign language recognition using LSTM and media pipe," *International Conference on Intelligent Computing and Control Systems,* ICICCS 2023, Madurai, India, May 17-19, 2023, pp. 1,086-1,091.

[13] M. Z. Uddin et al., "Real-time norwegian sign language recognition using mediapipe and LSTM," *Multimodal Technologies and Interaction,* vol. 9, no. 3, pp. 23-29, February 2025.

[14] N. Anithadevi et al., "MediaPipe-LSTM-enhanced framework for real-time dynamic sign language recognition in inclusive communication systems," *Engineering Reports,* vol. 7, no. 7, pp. 1-19, July 2025.

[15] B. A. Abdullah et al., "Advancements in sign language recognition : A comprehensive review and future prospects," *IEEE 6th International Conference on Advanced Computing,* IACC 2024, Bhimavaram, India, September 12, 2024, pp. 128,871-128,895.

[16] A. Khan et al., "Deep learning approaches for continuous sign language recognition : A comprehensive review," *IEEE Conference on Computer Vision and Pattern Recognition,* CVPR, Las Vegas, USA, April 27-30, 2025, pp. 2,818-2,826.

[17] T. Karanram and S. Srisuttiyakorn, "Development of thai sign language recognition model using deep learning for communication between student teachers and students with special needs : Application of CNN and RNN models," *Journal of Educational Measurement Mahasarakham University,* vol. 31, no. 1, pp. 152-172, June 2025.

[18] A. Chaikaew et al., "Thai sign language recognition : An application of deep neural network," *Joint International Conference on Digital Arts, Media and Technology,* ICLR 2021, San Diego, USA, March 3-6, 2021, pp. 128-131.

[19] Thai Sign Language Dictionary, *"Cute in Thai Sign Language,"* [Online]. Available: https://www.th-sl.com/word/9-095/. [Accessed : September 21, 2025].

[20] ASL Bloom, *"Cute in American Sign Language,"* [Online]. Available: https://www.aslbloom.com/signs/cute. [Accessed : September 21, 2025].

[21] Y. Fan, "The gesture recognition improvement of mediapipe model based on historical trajectory assist tracking, kalman filtering and smooth filtering," *International Conference on Information and Communication Technology Convergence*, CISAI 2024, Jeju, Korea (South), September 13-15, 2024, pp. 641-647.

[22] K. Myagila et al., "Framework for detecting and recognizing sign language using absolute pose estimation difference and deep learning," *Machine Learning with Applications*, vol. 21, no. 3, pp. 1-12, August 2025.

[23] T. S. Varshini and P. Rukmani, "MP-gestLSTM : Real time gesture detection using mediapipe and LSTM," *Systems Science & Control Engineering*, vol. 13, no. 1, pp. 1-18, November 2025.

[24] V. Ravikiran, "Real-time sign language recognition and translation using mediapipe and LSTM-based deep learning," *International Journal of Computer Applications*, vol. 187, no. 25, pp. 10-14, July 2025.

[25] Y. Meng et al., "Real-time hand gesture monitoring model based on mediapipe's registerable system," *Sensors*, vol. 24, no. 19, pp. 1-16, September 2024.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *KKU Engineering Journal*, vol. 9, no. 8, pp. 1,735-1,780, November 1997.

[27] C. Olah, *"Understanding LSTM Networks,"* [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed : September 22, 2025].

[28] D. Laines et al., "Isolated sign language recognition based on tree structure skeleton images," *Conference on Computer Vision and Pattern Recognition Workshops*, CVPRW 2023, Vancouver, Canada, June 17-24, 2023, pp. 276-284.

[29] J. Huang and V. Chouvatut, "Video-based sign language recognition via residual network and LSTM network," *Journal of Imaging*, vol. 10, no. 6, pp. 1-15, June 2024.

[30] W. Vijitkunsawat et al., "Deep multimodal-based finger spelling recognition for thai sign language : A new benchmark and model composition," *Machine Vision and Applications*, vol. 35, no. 4, pp. 76-168, May 2024.

[31] C. Guettas et al., “AI-driven sign language recognition system with NLP-enhanced transcription,” *Transactions on Computer and Information Technology*, vol. 19, no. 4, pp. 597-608, October 2025.

[32] Y. Han et al., “A study on the STGCN-LSTM sign language recognition model based on phonological features of sign language,” *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, Las Vegas, USA, April 27-29, 2025, pp. 74,811-74,820.

[33] M. Pu et al., “Siformer feature-isolated transformer for efficient skeleton-based sign language recognition,” *International Conference on Electronics Technology*, ICET, Melbourne, Australia, October 28-30, 2024, pp. 9,387-9,396.

[34] C. Luna-Jiménez et al., “Interpreting sign language recognition using transformers and mediapipe landmarks,” *International Conference on Image*, ICIVC, Paris, France, October 9-13, 2023, pp. 373-377.

[35] G. Sánchez-Brizuela et al., “Lightweight real-time hand segmentation leveraging mediaPipe landmark detection,” *Virtual Reality*, vol. 27, no. 5, pp. 3,125-3,132, September 2023.

[36] K. Lin et al., “Skeleton-based isolated sign language recognition with part mixing,” *Proceedings of the Second IEEE and ACM International Symposium on Mixed and Augmented Reality*, ISMAR Tokyo, Japan, October 10, 2023, pp. 205.

[37] A. Bhadouria et al., “LSTM-based recognition of sign language,” *International Conference in Distributed in Distributed Computing & Internet Technology*, ICDCIT 2014, Bhubaneswar, India, August 8-10, 2024, pp. 508-514.

[38] I. Rodríguez-Moreno et al., “Sign language recognition by means of common spatial patterns,” *Proceedings of the 2021 5th International Conference on Machine Learning and Soft Computing*, ICMLSC, Da Nang, Viet Nam, January 29-31, 2021, pp. 96-102.

[39] K. M. Dafnis et al., “Bidirectional skeleton-based isolated sign recognition using graph convolutional networks,” *Thirteenth Language Resources and Evaluation Conference*, LREC, Marseille, France, June 21-23, 2022, pp. 7,328-7,338.

[40] Y. Min et al., “A closer look at skeleton-based continuous sign language recognition,” *International Conference on Computer Vision Workshops,* ICCVW, Wuhan, China, October 19-23, 2025, pp. 4,968-4,974.

[41] Y. Rao et al., “Dynamic sign language recognition and translation through deep learning,” *Journal of Theoretical and Applied Information Technology,* vol. 102, no. 21, pp. 7,923-7,947, November 2024.

[42] S. Lata, *“The Automated Sign Language Translation System Using Deep Learning,”* B.Eng. Thesis (Automation Engineering), Mahasarakham University, Thailand, 2023.

[43] M. Gil-Martín et al., “Hand gesture recognition using mediapipe landmarks and deep learning networks,” *International Conference on Powder Metallurgy and Particulate Materials,* ICAART, Las Vegas, USA, February 23-25, 2025, pp. 24-30.

[44] A. Chaikaew, “An applied holistic landmark with deep learning for thai sign language recognition,” *International Technical Conference on Circuits/Systems, Computers and Communications,* ITC-CSCC, Phuket, Thailand, July 5-8, 2022, pp. 1,046-1,049.

[45] J. Li et al., “A continuous sign language recognition method based on temporal modeling and semantic alignment,” *International Conference on Learning Representations,* ICLR 2025, Jinan, China, November 14-16, 2025, pp. 170-174.

[46] P. Nakjai et al., “Thai finger spelling localization and classification under complex background using a yolo-based deep learning,” *International Conference on Computer Modeling and Simulation,* ICCMS, New York, USA, North Rockhampton, Australia, January 16-19, 2019, pp. 230-233.

[47] H. Yoo et al., “Real-time dynamic sign language recognition using LSTM based on mediapipe hand data,” *International Conference on Consumer Electronics-Taiwan,* ICCE, PingTung, Taiwan, July 17-19, 2023, pp. 17-18.

[48] Y. Zhang et al., “Sign language recognition based on CNN-BiLSTM using RF signals,” *IEEE Sensors Journal,* vol. 24, no. 2, pp. 1,456-1,467, December 2024.

[49] S. Shinde et al., “Sign language recognition using deep learning,” *International Conference on Computing Communication and Networking Technologies*, ICCCNT, Mandi, India, June 18-22, 2024, pp. 1-5.

[50] Y. Zhang and X. Jiang, “Recent advances on deep learning for sign language recognition,” *Computer Modeling in Engineering & Sciences*, vol. 139, no. 3, pp. 2,399-2,450, March 2024.

[51] M. Recalde et al., "Creating an accessible future: developing a sign language to speech translation mobile application with mediapipe hands technology,” *International Conference on ICT for Smart Society*, ICISS, Bandung, Indonesia, September 6-7, 2023, pp. 558-563.

[52] J. P. Sahoo et al., “Real-time hand gesture recognition using fine-tuned convolutional neural network,” *International Journal of Fatigue*, vol. 22, no. 3, pp. 706-719, January 2022.

Appendices

Appendix A.
Symbol Definitions for Mathematical Feature Formulations

Table A.1 Landmark and Coordinate Variables

Symbol	Meaning
$\mathbf{p}_{j,t}$	Three-dimensional coordinate of hand landmark j at time frame t
$\hat{\mathbf{p}}_{j,t}$	Normalized coordinate of landmark j at time t
$\mathbf{w}_t$	Wrist landmark coordinate at time t
$\mathbf{m}_t$	Middle finger MCP landmark coordinate at time t
$\mathbf{p}_{tip,f}(t)$	Fingertip landmark coordinate of finger f at time t
$\mathbf{p}_{wrist,f}(t)$	Wrist landmark coordinate at time t
$\mathbf{p}_{index\_MCP}$	MCP landmark of the index finger
$\mathbf{p}_{pinky\_MCP}$	MCP landmark of the pinky finger
$\mathbf{p}_{body}$	Upper-body reference landmark (e.g., nose or shoulder joints)

Table A.2 Distance and Length Measurements

Symbol	Meaning
$d_f(t)$	Fingertip-to-wrist distance for finger f
$d_{hb}(t)$	Distance between wrist and upper-body reference point
$L_i(t)$	Finger length approximation for finger i
$s_k(t)$	Distance between adjacent fingertips representing finger spread

Table A.3 Angle, Orientation and Temporal Motion Variables

Symbol	Meaning
θ	Joint angle formed by two adjacent bone vectors
$\theta_{i(t)}$	Finger joint angle of finger i at time t
θ_{thumb}	Joint angle used to determine thumb extension
$\vec{v}_1, \vec{v}_2$	Palm direction vectors constructed from wrist to MCP joints
$n(t)$	Normal vector representing palm orientation
$\Delta Pose_t$	Temporal difference between normalized arm pose features
$PoseNorm_{(t-1)}$	Pose feature vector from previous frame

Table A.4 Binary State Indicators

Symbol	Meaning
$Ext_f(t)$	Finger extension flag indicating whether finger f is extended
$Bent_i(t)$	Finger bending flag indicating whether finger i is bent
$ThumbExtended$	Binary thumb posture indicator
$b_h(t)$	Hand presence flag indicating whether a hand is detected

The symbols defined above describe geometric, kinematic, and temporal characteristics of hand and arm movements extracted from skeletal landmarks. These definitions ensure that the mathematical formulations used for feature engineering are consistent with the implementation pipeline and can be reproduced across different datasets and deployment environments.