Case Study

Using LSTM to translate Thai sign language to text in real time

Werapat Jintanachaiwat[1] · Kritsana Jongsathitphaibul[1] · Nopparoek Pimsan[1] · Mintra Sojiphan[1] · Amorn Tayakee[1] · Traithep Junthep[1] · Thitirat Siriborvornratanakul[1]

Received: 26 September 2023 / Accepted: 18 February 2024
Published online: 29 February 2024

Abstract
Between 2019 and 2022, as the Covid-19 pandemic unfolded, numerous countries implemented lockdown policies, leading most corporate companies to permit employees to work from home. Communication and meetings transitioned to online platforms, replacing face-to-face interactions. This shift posed challenges for deaf or hearing-impaired individuals who rely on sign language, using hand gestures for communication. However, it also affected those who can hear clearly but lack knowledge of sign language. Unfortunately, many online meeting platforms lack sign language translation features. This study addresses this issue, focusing on Thai sign language. The objective is to develop a model capable of translating Thai sign language in real-time. The Long Short-Term Memory (LSTM) architecture is employed in conjunction with MediaPipe Holistic for data collection. MediaPipe Holistic captures keypoints of hand, pose, and head, while the LSTM model translates hand gestures into a sequence of words. The model's efficiency is assessed based on accuracy, with real-time testing achieving an 86% accuracy, slightly lower than the performance on the test dataset. Nonetheless, there is room for improvement, such as expanding the dataset by collecting data from diverse individuals, employing data augmentation techniques, and incorporating an attention mechanism to enhance model accuracy.

1 Introduction

The situation of people with disabilities in Thailand as of September 30, 2023 (reported on the website as of November 9, 2023) [17] reveals that the three highest-ranking types of disabilities are as follows: 51.57% (1,155,339 individuals) have mobility/physical disabilities, 18.57% (415,999 individuals) have hearing/speech disabilities, and 8.24% (184,542 individuals) have visual impairments. Notably, it is evident that individuals with hearing impairments, who rely on sign language, constitute the second-highest number among these categories. Effective communication with deaf or hearing-impaired individuals has been a significant challenge, especially for those unfamiliar with sign language. Thus, the research in the field of Sign Language Translation (SLT) has become popular [21] and it is very valuable.

Sign language can be categorized into two main types. First is gesture language, which utilizes hand motions and facial expressions to convey specific messages. Second is finger-spelling, employing hand signs to represent letters or numerals for word spelling. Thai sign language has been influenced by international sign languages prevalent in Canada, the United States of America, West Africa, and Southeast Asia. Each country has its own unique sign language, and even within the same country, signs may vary based on the user's age and the region where the sign language is employed. With the advent of the Covid-19 pandemic, online meetings have become a crucial tool and are expected to become

✉ Thitirat Siriborvornratanakul, thitirat@as.nida.ac.th; Werapat Jintanachaiwat, werapat.jin@stu.nida.ac.th; Kritsana Jongsathitphaibul, kritsana.jon@stu.nida.ac.th; Nopparoek Pimsan, nopparoek.pim@stu.nida.ac.th; Mintra Sojiphan, mintra.soj@stu.nida.ac.th; Amorn Tayakee, amorn.tay@stu.nida.ac.th; Traithep Junthep, traithep.jun@stu.nida.ac.th | [1]Graduate School of Applied Statistics, National Institute of Development Administration, 148 SeriThai Rd., Bangkapi, 10240 Bangkok, Thailand.

the new norm. However, individuals with hearing impairments may encounter difficulties participating in such meetings. Our research will concentrate on gesture language, which is more suitable for simple conversations, as opposed to finger-spelling, which is predominantly used for name spelling or educational purposes.

In recent times, numerous research projects have emerged focusing on sign language translation, proposing various methods and techniques. These include Machine Learning (ML) models, Neural Networks such as Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and hierarchical-LSTM (HLSTM). Many studies aim to develop models capable of interpreting fundamental gestures in sign language captured in real-time by cameras and translating them into text or audio. However, the most significant challenge faced by sign language translation models is their effectiveness in real-world scenarios. Specifically, factors such as environmental conditions during input capture (e.g., lighting, background, and camera position), occlusion (e.g., hand gestures moving out of the frame), and sign boundary detection can impact the accuracy of the model. In this research, we are seeking a model that excels in real-time translation, demonstrating high accuracy in continuous conversation scenarios.

2 Related works

There are numerous studies on techniques related to sign language translation, and alphabet detection in sign language is a notable area of focus. Unlike methods involving gloves or sensors, the approach presented in [1] leverages image processing techniques for letter recognition in American sign language. Initially, the images undergo conversion into a binary format to isolate hands from the background, followed by contour detection to identify the hand's outline. Points along the contour are extracted, and the Convex Hull technique is employed to distinguish concave and convex areas, thereby identifying fingers and the spaces between them. Different groups of letters necessitate distinct identification techniques; some may rely solely on contour area, while others may require a combination of parameters. For example, the identification of letters such as D, J, H, I, and U involves considering parameters such as solidity, aspect ratio, and angles.

Before deep learning gained popularity, image classification and feature extraction could be achieved through traditional machine learning. However, its application nowadays may be limited to demonstration purposes, as it consumes significantly fewer resources compared to deep learning techniques. Nevertheless, it may involve numerous manual thresholds and complexities. The process of manual feature extraction starts with converting images into feature vectors, and the vector representation is then used as input for further analysis. The work of [2] introduced sign language translation using multidimensional Hidden Markov Models (HMM), utilizing data from a sensory glove to identify hand shape and motion tracking. HMM extracted the constituent signs implicitly in multiple dimensions through a stochastic process, allowing for interactive learning and recognition. In [3], the Support Vector Machine (SVM) was applied with Hidden Conditional Random Fields to enhance sign recognition without the aid of gloves or wearable tracking devices. Feature extraction was classified into two layers, with the static gesture recognition layer serving as a trained classifier extracted by SVM. The dynamic gesture recognition layer, encompassing discrete and dynamic gesture patterns, was applied by Hidden Conditional Random Fields to consider gesture sequences through emission probabilities. This two-layer application proves useful for linguistic information as a discriminative model, improving system performance without overfitting.

Authors in [4] developed a sign language translation model using image processing and traditional machine learning techniques to translate Indian Sign Language to English. The proposed system uses hand gestures as input and generates audio as output. Hand gestures are captured by a video camera, and captured videos are divided into frames. Image processing techniques, including hand segmentation using the YCbCr color space and histograms of oriented gradients (HOG) for feature extraction, are employed. Classification and recognition stages are achieved through SVM. Finally, the system returns output text, and Google Text to Speech transforms the output text into audio. In the work of [5], real-time Myanmar sign language recognition is studied to translate 30 sign gestures from videos. Hand detection is processed by converting frames into the YCbCr color space for skin detection through thresholding, followed by close and open morphological operations. Background removal is performed using hole-filling operations. The hand frame is then cropped and converted into grayscale. Feature extraction is processed using Principal Component Analysis (PCA), and SVM is applied to classify sign language. The SVM algorithm is trained using 3,000 hand signal images, achieving an accuracy rate for each sign gesture ranging from 80% to 100%. The work of [6] introduces discrete square-shaped Haar wavelet transform, including Haar local binary pattern (LBP) features for decomposition. A 2D point cloud is developed to represent posture as feature extraction. Global and local features are calculated to represent the signer's hand in video sequences. The multi-class multi-label Adaboost algorithm is applied for pattern recognition and queries the signs in the video.

After the evolution of computer and computation technology, deep neural networks have become increasingly powerful. Numerous researchers in SLT have explored neural network techniques to enhance accuracy. In the work of [7], SLT by CNN is proposed, incorporating hand detection and classification. This paper integrates Single Shot MultiBox Detection (SSD), Inception v3, and SVM on fragments of American sign language (ASL) finger-spelling. The simulation results demonstrate a high accuracy rate. However, the study relies on an isolated image dataset and focuses solely on ASL finger-spelling alphabet translation. To enhance real-world applicability in communication, there is a need for real-time SLT. The work of [8] introduces real-time ASL recognition with CNN, utilizing transfer learning on the pre-trained GoogLeNet architecture and testing with real-time users. This research exhibits strong performance in real-time finger-spelling alphabet translation. Many researchers have primarily focused on isolated finger-spelling alphabets, which may not be suitable for real-time communication. In [9], experiments on SLT to words and sentences are conducted using CNN and LSTM. The study reveals that CNN performs well for isolated sign language recognition, while LSTM excels in continuous word recognition. However, the study indicates a 72% accuracy for LSTM, leaving room for improvement in model accuracy. The work of [10] introduces a Hierarchical Long Short-Term Memory (HLSTM) framework designed to capture continuous sign language and translation in sequential order. Through experiments with various models, it was observed that both HLSTM and HLSTM-attention achieved an accuracy above 90% in continuous conversation scenarios.

Recent research published in 2023 [18–20] reveals that there is currently no standard scheme or foundational models for interpreting continuous sign language. Consequently, our study will utilize RNN/Bidirectional RNN, LSTM/Bidirectional LSTM, and FNN-LSTM for real-time translation of Thai sign language to Thai language text during continuous conversation. Instead of relying on conventional image processing methods for object detection or deep learning-based object detection [23], this paper will demonstrate the use of the MediaPipe framework to gather key points for extracting left hand, right hand, face, and posture landmarks. Subsequently, we will classify and train Sign Language Translation (SLT) using a traditional neural network model, particularly within the RNN family, applied to a Thai sign language dataset. Our aim is to simplify the complexity of existing methodologies by leveraging MediaPipe and the LSTM model to achieve a high-performance model suitable for real-time continuous conversation.

3 Proposed methods

In our process design (see Fig. 1), the webcam is activated to capture every sign language gesture. Each frame generates keypoints using the MediaPipe Holistic model [11]. Once the keypoints are collected, they are sequentially fed into the deep learning model. The trained model then predicts the probability of each word. Finally, the result of sign language translation is determined by selecting the word with the highest probability exceeding a specified threshold.

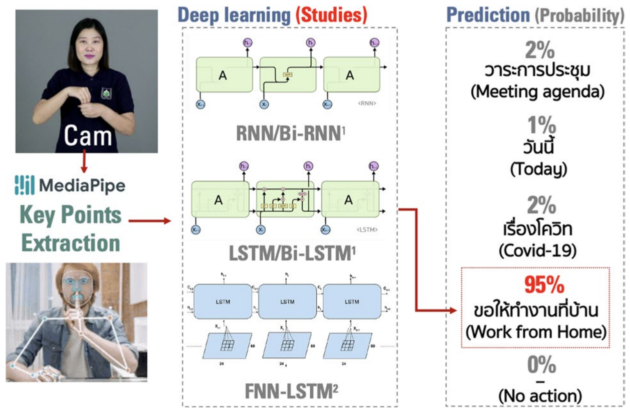

Fig. 1 The overview workflow of this study. MediaPipe framework is used to extract keypoints from sign gestures. The dataset feed to the five models to predict probability of hand gestures. (Human and model images were collected from www.google.com)

3.1 Data acquisition and pre-processing

In this paper, MediaPipe Holistic is utilized to collect keypoints from the hand, arm, body, and face. MediaPipe provides landmark detection and segmentation on all frames [12]. The acquired keypoints are then saved as a frame, representing a sequence of events for a particular sign. The MediaPipe Holistic pipeline integrates models for pose, face, and hand detection, namely MediaPipe Pose, MediaPipe Face Mesh, and MediaPipe Hands. Each model requires different input specifications; for example, the pose estimation model requires a lower image resolution than the face and hand models. MediaPipe Holistic utilizes these models to generate a total of 543 landmarks (33 pose landmarks, 468 face landmarks, and 21 hand landmarks per hand).

To collect data for the training set and testing set, the process begins by setting up video capture to extract frames using OpenCV. Next, MediaPipe Holistic is constructed to obtain the holistic model used for detection, and MediaPipe drawing utilities are employed to draw keypoints on the hand, pose, and face. Creating a detection with MediaPipe involves converting the collected image from BGR to RGB, setting it as unwritable, performing the detection, setting it back to writable, and transforming it from RGB back to BGR. This is necessary because the feeds from OpenCV are in BGR format, but MediaPipe performs detection in RGB format. The MediaPipe Holistic model operates by making an initial detection and then tracking the keypoints. The MediaPipe draw landmark function visualizes the landmarks in real-time, illustrating the connections between the hand, pose, and head. The keypoint values are then extracted and saved in a Numpy array, which consists of a series of 30 arrays, each containing 1662 values. A single frame comprises 1662 landmark values stored in each of the 30 arrays. Subsequently, folders are established to collect arrays as part of our sign language dataset. Each action comprises 30 distinct frames or 30 different sets of keypoints, and for each action, 30 videos were gathered. For this study, the dataset was split into a 90%-10% train-test ratio. The dataset employed in this project is outlined in Table 1. The selection of words and sentences in the table was based on their ease of performance in sign language, avoiding excessive complexity and length.

3.2 Experimental models

In this work, we aim to identify the most suitable model for training on hand, face, and pose gesture keypoints extracted by MediaPipe Holistic. We are experimenting with five models, namely RNN/Bidirectional-RNN, LSTM/Bidirectional-LSTM, and FNN-LSTM. All hyperparameters presented in this section are the best outcomes achieved through our trial-and-error experiments.

3.2.1 Recurrent neural network (RNN)

The objective of this study is to predict the meaning of hand gestures from a sequence of frames. We propose implementing Recurrent Neural Networks (RNN), as they are well-suited for handling sequential tasks. As mentioned in [13], the RNN architecture is good at handling temporal tasks by referencing inputs from previous stages and calculating them using the same function. Equations regarding the RNN in Fig. 2 are:

$$s_t = f(s_{t-1} * W + x_t * U + b_h) \tag{1}$$

$$y_t = s_t * V + b_y \tag{2}$$

s_t is the vector resulting from the output of the hidden layer function f. It is determined by the combination of the output vector from the previous state, s_{t-1}, multiplied by the internal weight W, and the current state input, x_t, multiplied by the

Table 1 Our sign language dataset

No.	Words	The number of videos	
		Train data	Test data
1	Meeting agenda	27	3
2	Today	27	3
3	About Covid-19	27	3
4	Please work from home	27	3
5	Null (no action)	27	3

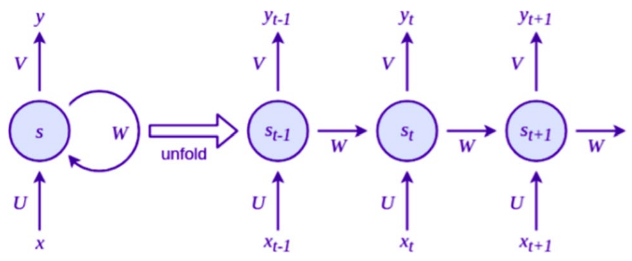

Fig. 2 The internal mechanism of RNN. (Source: [14])

weight V, along with the bias of the hidden layer. s_t is then multiplied by the weight V and combined with the bias b_y. The ultimate output of the RNN is denoted as y_t. s_t is subsequently passed on to the next state at $t + 1$, while the weights U, W, and V are shared.

In the data preparation process, the data is represented in the form of a NumPy array with dimensions of 30 x 1662. Consequently, the multi-layer RNN architecture depicted in Fig. 3 comprises five hidden state layers. The first layer accommodates the input shape of 30 x 1662. The initial three layers consist of 64, 128, and 64 nodes, respectively, with the ReLU activation function. The subsequent two dense layers consist of 64 and 32 nodes, employing ReLU as the activation function. Finally, Softmax is applied to the output of the final layer. The model is compiled with Adam as the optimizer and trained for 300 epochs.

3.2.2 Bidirectional RNN (Bi-RNN)

The Bi-RNN architecture depicted in Fig. 4 supports data processing in both the forward and backward directions with separate hidden layers. Ultimately, the outputs from both directions are fed into the same output layer. Since the data is sequential, Bi-RNN can effectively handle such sequential data. The equations for the Bi-RNN are as follows:

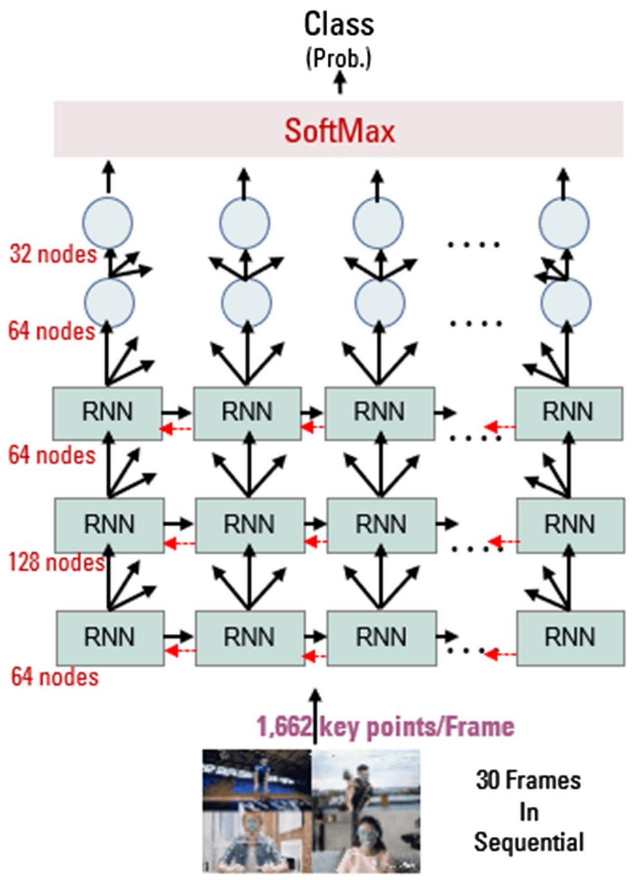

Fig. 3 The architecture employed in this study involves RNN and Bidirectional RNN. The three RNN layers are connected to two dense layers to calculate the probability of hand gesture meanings using the Softmax function

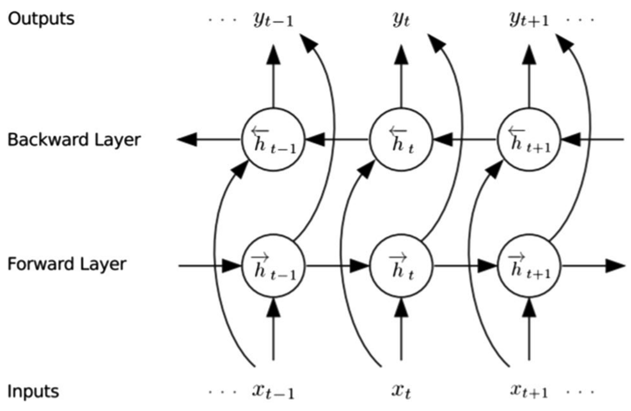

Fig. 4 The internal mechanism of Bidirectional RNN. (Source: [15])

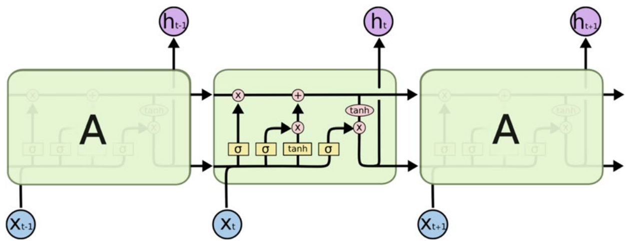

Fig. 5 The internal mechanism of LSTM. (Source: [16])

$$h^{\rightarrow} = H(W_{xh^{\rightarrow}} x_t + W_{h^{\rightarrow}h^{\rightarrow}} h^{\rightarrow}{}_{t-1}) + b_{h^{\rightarrow}} \tag{3}$$

$$h^{\leftarrow} = H(W_{xh^{\leftarrow}} x_t + W_{h^{\leftarrow}h^{\leftarrow}} h^{\leftarrow}{}_{t+1}) + b_{h^{\leftarrow}} \tag{4}$$

$$y_t = W_{h^{\rightarrow}y} h^{\rightarrow}{}_t + W_{h^{\leftarrow}y} h^{\leftarrow}{}_t + b_y \tag{5}$$

Based on [13], the architecture of Bi-RNN is similar to that of RNN as shown in Fig. 3. The weights and input data from both the current and previous states are multiplied and combined by adding the bias b_h. After computing the forward and backward hidden layers, the output data is fed to the same output layer, multiplied by the weights, and added to the bias b_y to yield the final result y_t. In our study, the structure of Bi-RNN is the same as RNN, as the efficiency between the models can be comparable. Therefore, the Bi-RNN consists of five hidden layers. The first layer supports the input shape of 30 x 1662. The initial three layers comprise 64, 128, and 64 nodes, respectively, applying ReLU as the activation function. The subsequent two dense layers consist of 64 and 32 nodes, employing ReLU as the activation function. Finally, Softmax is applied to the output of the final layer. The model is compiled with Adam as the optimizer and trained for 300 epochs.

3.2.3 Long short term memory (LSTM) and bidirectional LSTM (Bi-LSTM)

Moreover, to enhance the capability of the recurrent model, this study also conducted experiments with the LSTM model to identify the best fit for continuous sign language, which is fed by 30 frames sequentially for each translation prediction. The equations for LSTM, as illustrated in Fig. 5, are as follows, where *U*, *W* are trainable weight, *b* are bias, *i* are input gate, *f* are forget gate, *o* are output gate, *g* are hidden state, c_t are internal storage, s_t are output information, and σ are sigmoid function:

$$i = \sigma(U^i x_t + W^i s_{t-1} + b_i) \tag{6}$$

$$f = \sigma(U^f x_t + W^f s_{t-1} + b_f) \tag{7}$$

$$o = \sigma(U^o x_t + W^o s_{t-1} + b_o) \tag{8}$$

$$g = tanh(U^g x_t + W^g s_{t-1} + b_g) \tag{9}$$

$$c_t = c_{t-1} \cdot f + g \cdot i \tag{10}$$

$$s_t = tanh(c_t) \cdot o \tag{11}$$

The equations for Bi-LSTM depicted in Fig. 6 are shown in Eq. 12–14, where Eq. 12 represents the forward route, and Eq. 13 represents the backward route; y_i is the output of the Bi-LSTM obtained by combining the results from both directional routes. The architecture of LSTM/Bi-LSTM for this study, is similar to the RNN structures mentioned in Fig. 3. The difference lies in the first three layers, which are changed from RNN to LSTM, while the rest remain the same.

$$h_i^1 = f(U^1 \cdot x_i + W^1 \cdot h_{t-1}) \tag{12}$$

$$h_i^2 = f(U^2 \cdot x_i + W^2 \cdot h_{i-1}) \tag{13}$$

$$y_i = softmax(V \cdot [h_i^1; h_i^2]) \tag{14}$$

3.2.4 Feed forward neural network with LSTM (FNN-LSTM)

In the architecture depicted in Fig. 7, the Feed-Forward Neural Network (FNN) is employed for feature extraction after receiving keypoint features input from MediaPipe as a vector with dimensions of 30 x 1662 for each frame. This is done to recognize local patterns in the sequence for pattern learning [16]. In this study, two FNNs with fully connected layers are utilized, featuring 128 x 64 nodes in the initialization layer and hidden layers, respectively. These layers are designated to classify and recognize local patterns in each sequence of frames before feeding the weighted output to the subsequent LSTM networks for sequential recognition. In the part of sequence recognition, this study applies a RNN with LSTM units. The LSTM consists of two layers, the first with 128 LSTM units and the second with 64 LSTM units, followed by a feed-forward fully connected layer as a dense layer to downsample to categorical output with Softmax activation. The ADAM algorithm is employed as a stochastic optimizer to minimize the categorical cross-entropy loss function.

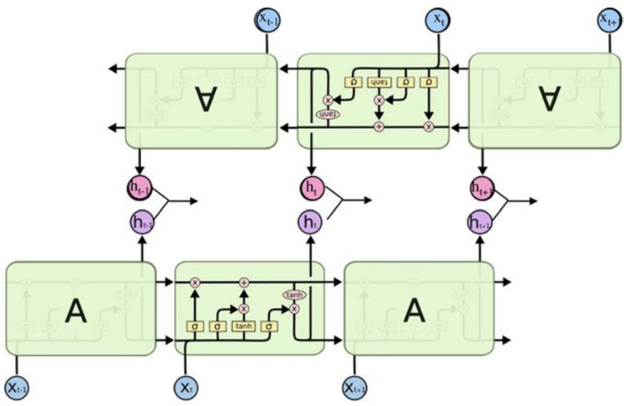

Fig. 6 The internal mechanism of Bidirectional LSTM. (Source: [16])

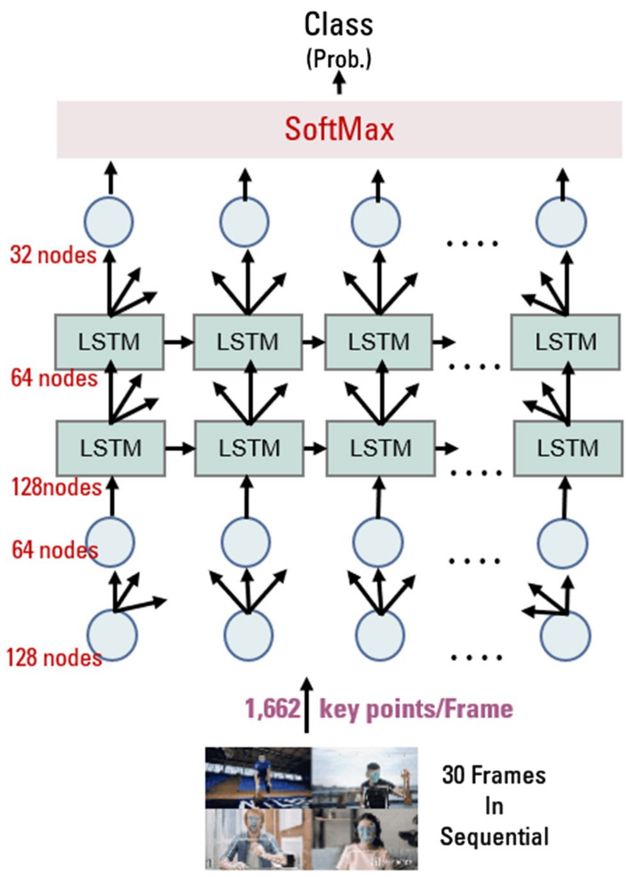

Fig. 7 The architecture of FNN-LSTM used in this study. The two FNN layers are connected to the two LSTM layers to calculate the probability of hand gesture meanings using the Softmax function

4 Experimental results and discussion

In this work, MediaPipe Holistic is chosen for feature extraction, and three main models are employed: RNN/Bidirectional RNN, LSTM/Bidirectional LSTM, and FNN-LSTM due to their capacity to capture sequential actions in continuous sign language. The selection of layers and nodes was based on the experimental results of each model. Five classes were chosen from thirteen Thai words commonly used in meetings during the Covid-19 situation. These words include "meeting agenda," "today," "Covid-19," "work from home," and the null action used to conclude a statement. These selected words represent common sentences and uncomplicated postures for data collection. During the experiment, recognized continuous sign postures will be displayed as text when the calculated probability exceeds the desired threshold of 0.95, while unrecognized signs will not be translated into any text. The chosen threshold was determined through experimentation, and lowering the threshold resulted in too much interference.

The detailed experimental results are presented in Fig. 8 and Table 2, where the "Model Accuracy" column represents the prediction accuracy on the test dataset, and the "Real-time Test Accuracy" column indicates the prediction accuracy during real-time testing. According to the table, LSTM yielded the best results on the test dataset with a prediction accuracy of 100%. However, when tested in real-time, the accuracy of LSTM dropped to 86%. RNN also achieved 100% accuracy on the test set but showed a lower performance during real-time testing, yielding a result of 64%. On the other hand, Bi-LSTM proved unsuitable for continuous Thai sign language due to the complexity of sequential recognition.

In continuous sign language, specific words are conveyed not only through hand signs but also involve facial expressions and body posture movements. In such cases, MediaPipe Holistic proves to be a suitable tool for this task. Utilizing MediaPipe Holistic instead of traditional CNN for feature extraction helps reduce programming complexity and provides valuable information for deeper analysis through deep learning. Nevertheless, using a single signer for data collection in our prototype system introduces the inevitable challenge of overfitting due to the limited dataset. Additionally, there may be issues during model execution by others who must replicate the same posture as the person who initially trained the dataset.

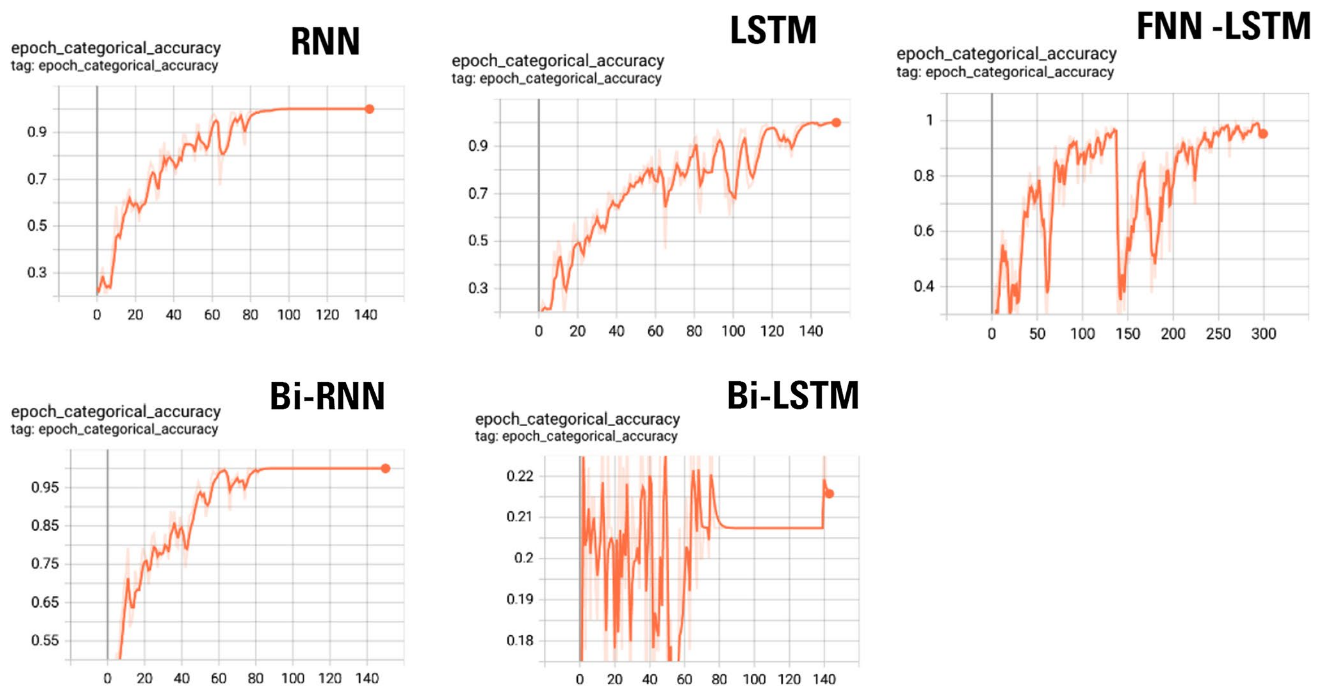

Fig. 8 Training accuracy of all experimental models

Table 2 Model evaluation

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 font indicates the best model

In the context of real-time sign language interpretation, the ability of a system to interpret continuous sign language promptly is crucial. However, the current prototype system, which relies on a single signer, has not undergone explicit testing for real-time performance. While MediaPipe is recognized for its rapid and real-time processing capabilities, it is important to acknowledge that deep learning models, despite their efficiency, are associated with substantial computational resource consumption. This characteristic may potentially introduce delays in achieving real-time interpretation. Consequently, future iterations of the system should include comprehensive testing under varying network conditions to assess its performance in scenarios where data transfer rates are low due to network issues. This will provide a more comprehensive understanding of the system's real-time interpretative capabilities, especially in adverse network conditions.

5 Conclusion and future works

In this paper involving only train-from-scratch basic neural networks, it is evident that the LSTM model stands out as the best performer in this study. However, its tendency to overfit arises due to the limited dataset derived from an individual subject, resulting in insufficient data for our self-collected dataset. Consequently, during real-world testing with diverse signers, the accuracy experiences a significant drop. Nevertheless, there exists room for enhancement in future studies, particularly in data collection. To address this, it is recommended to improve both the quantity and quality of the dataset by gathering more diverse data from various signers and applying data augmentation techniques to generate fresh and varied instances for the dataset. This approach can substantially improve the model's performance and accuracy when the dataset is both rich and comprehensive. Additionally, more evaluation metrics apart from accuracy should be employed to ensure a comprehensive evaluation of these black-box neural networks, particularly in the consistency aspect across the whole interpretation session [24]. Furthermore, incorporating matrix operation techniques commonly used in deep learning to reposition the coordinates of points can aid in boosting accuracy for different signers, distances, and camera angles. Attention mechanisms, fine-tuning model hyperparameters, and alternative temporal-based model architectures [22, 25] can also contribute to refining model accuracy.

Author contributions All authors contributed equally to this manuscript.

Funding This research received no specific Grant from any funding agency in the public, commercial, or not-for-profit sectors.

Data availability The datasets generated and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Declarations

Consent to participate Not available as the results presented by this research do not involve any identifiable personal data.

Competing interests The authors declare that they have no competing interests.

Appendix

Software Libraries:

1) Python version 3.7.4
2) Numpy version 1.19.5
3) Pandas version 1.2.3
4) OpenCV2 version 4.5.3
5) MediaPipe version 0.8.9.1
6) Keras Tensorflow version 2.7.0
7) Mac OS Big Sur Version 11.5.2

Hardware resources:

1) MacBook Pro
2) Processor 2.4 GHz Quad-Core Intel Core i5
3) Memory 8 GB 2133 MHz LPDDR3
4) Graphics Intel Iris Plus Graphics 655 1536 MB

References

1. K. Manikandan, A. Patidar, P. Walia, AB. Roy, "Hand gesture detection and conversion to speech and text." International Conference on Innovations and Discoveries in Science, Engineering and Technology(ICIDSET), 2018.
2. Wang H, Leu MC, Oz C. American sign language recognition using multi-dimensional hidden Markov models. J Inf Sci Eng. 2006;22(5):1109–23.

3. Souza CR, Pizzolato EB. Sign language recognition with support vector machine and hidden conditional random fields: going from fingerspelling to natural articulated words. Machine learning and data mining in pattern recognition, lecture notes in computer science. 2013; 84–98.
4. Vedak O, Zavre P, Todkar A, Patil M. Sign language interpreter using image processing and machine learning. Int Res J Eng Technol (IRJET). 2019;6(4).
5. Tun M, Lwin T. Real-time Myanmar sign language recognition system using PCA and SVM. Int J Trends Sci Res Dev (IJTSRD). 2019;3(5):2361–6.
6. Kumar PP, Reddy PVGDP, Rao PS. "SIGN LANGUAGE RECOGNITION WITH MULTI FEATURE FUSION AND ADABOOST CLASSIFIER." ARPN J Eng Appl Sci, 13(4), 2018, pp. 1410-1419.
7. Abiyev RH, Arslan M, Idoko JB. "Sign Language Translation Using Deep Convolutional Neural Networks." KSII Transactions on Internet and Information Systems (TIIS), 14(2), 2020; pp. 631-653.
8. Garcia B, Viesca SA. "Real-time American Sign Language recognition with Convolutional Neural Networks." CS231n: convolutional Neural Networks for Visual Recognition, Stanford University - Spring 2016 student report, 2016; http://cs231n.stanford.edu/reports/2016/pdfs/214_Report.pdf
9. Elhagry A, Elrayes R. "Egyptian sign language recognition using CNN and LSTM." arXiv, 2021; https://arxiv.org/abs/2107.13647.
10. Guo D, Zhou W, Li H, Wang M. "Hierarchical LSTM for Sign Language Translation." Proceedings of the International AAAI Conference on Web and Social Media, 32(1), 2018;
11. google/mediapipe, "Github-google/mediapipe", https://github.com/google/mediapipe.
12. Lugaresi C, Tang J, Nash H, McClanahan C, Uboweja E, Hays M, Zhang F, Chang C, Yong MG, Lee J, Chang W, Hua W, Georg M, Grundmann M. "MediaPipe: a framework for building perception pipelines." arXiv, 2019; https://arxiv.org/abs/1906.08172.
13. Domenech A. "SIGN LANGUAGE RECOGNITION: ASL Recognition with MediaPipe and Recurrent Neural Networks." Bachelor-Thesis, FH Aachen University of Applied Sciences, 2020; https://upcommons.upc.edu/bitstream/handle/2117/343984/ASL
14. Vasilev I, Slater D, Spacagna G, Roelants P, Zocca V. "Python deep learning: exploring deep learning techniques and neural network architectures with Pytorch, Keras, and TensorFlow." Packt Publishing Ltd, 2019;
15. Graves A, Jaitly N, Mohamed A. "Hybrid speech recognition with deep bidirectional LSTM." IEEE workshop on automatic speech recognition and understanding, 2013; pp. 273-278.
16. Colah blog, "Understanding LSTM Networks." Posted on August 27, 2015, https://colah.github.io/posts/2015-08-Understanding-LSTMs/.
17. Department of Empowerment of Persons with Disabilities, "The situation of people with disabilities on September 230, 2023." Posted on November 9, 2023., https://dep.go.th/th/law-academic/knowledge-base/disabled-person-situation/
18. Sreemathy R, Turuk MP, Chaudhary S, Lavate K, Ushire A, Khurana S. Continuous word level sign language recognition using an expert system based on machine learning. Int J Cogn Comput Eng. 2023;4:170–8.
19. Hu L, Gao L, Liu Z, Feng W. "Continuous sign language recognition with correleation network." IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023; pp. 2529-2539.
20. Pathan RK, Biswas M, Yasmin S, Khandaker MU, Salman M, Youssef AAF. "Sign language recognition using the fusion of image and hand landmarks through multi-headed convolutional neural network." Scientific Reports, 13, 2013;
21. Adeyanju IA, Bello OO, Adegboye MA. "Machine learning methods for sign language recognition: a critical review and analysis." Intelligent Systems with Applications, 12, 2021;
22. Guo L, Xue W, Guo Q, Liu B, Zhang K, Yuan T, Chen S. "Distilling Cross-Temporal Contexts for Continuous Sign Language Recognition." IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10771-10780, 2023;
23. Attia NF, Ahmed MTFS, Alshewimy MAM. "Efficient deep learning models based on tension techniques for sign language recognition." Intelligent systems with applications, 20, 2023;
24. Zuo R, Mak B. Improving Continuous Sign Language Recognition with Consistency Constraints and Signer Removal. ACM Transactions on Multimedia Computing: Communications, and Applications; 2024.
25. Liu Y, Nand P, Hossain MA, Nguyen M, Yan WQ. "Sign language recognition from digital videos using feature pyramid network with detection transformer." Multimedia Tools and Applications, 82, pp. 21673-21685, 2023;