J Inf Process Syst, Vol.21, No.4, pp.427~438, August 2025
https://doi.org/10.3745/JIPS.04.0356 ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Korean Sign Language Recognition Using LSTM and Video Datasets

Soo-Yeon Jeong[1], Ho-Yeon Jeong[2], and Sun-Young Ihm[3,*]

Abstract

Deaf individuals primarily use sign language, which consists of hand gestures and body movements, as their main means of communication. It is difficult for non-disabled people to understand the visual form of sign language, and sign language recognition technology is required to facilitate communication. However, unlike spoken languages used by the general population, sign languages take a visual form and can be recognized through video or image data before being translated into other languages. In this study, we proposed a Korean sign language recognition and sentence conversion system based on long short-term memory (LSTM) using video datasets. To build a Korean sign language dataset, we automatically collected and preprocessed video data of sign language gestures, which were then used as input for the LSTM model. LSTM has strengths in processing sequential data and can effectively recognize the sequences and patterns of sign language gestures. The experimental results measured the accuracy of the model and analyzed its performance based on sign language gesture recognition and display. This study confirmed the effectiveness of the proposed approach and is expected to contribute to the advancement of Korean sign language recognition technology.

Keywords

Korean Sign Language Recognition, Long Short-Term Memory (LSTM), Sentence Conversion, Video Recognition

1. Introduction

Sign language is used by deaf people for communication, but they still experience communication difficulties in their daily lives. Although sign language has been recognized as a language, its translation development is slower than that of other languages conveyed through text, as sign language is expressed through hand gestures and movements. Therefore, a system that recognizes and interprets sign language is necessary to facilitate smooth communication among sign language users. A sign language recognition system provides opportunities for communication between deaf and non-deaf individuals, thereby offering deaf individuals more opportunities to participate in their daily lives. Moreover, as communication with others becomes smoother, deaf individuals demonstrate their abilities in various fields. With the advancement of deep learning, the recognition technology for videos and images has progressed and is being applied to various studies on sign language recognition [1,2].

Manuscript received September 26, 2024; first revision December 23, 2024; second revision February 5, 2025; accepted March 2, 2025.
***Corresponding Author:** Sun-Young Ihm (sunnyihm@pcu.ac.kr)
[1] Division of Software Engineering, Pai Chai University, Daejeon, Korea (sy.jeong@pcu.ac.kr)
[2] Department of Artificial Intelligence, Kyung Hee University, Yongin, Korea (2580jhy@naver.com)
[3] Department of Computer Engineering, Pai Chai University, Daejeon, Korea (sunnyihm@pcu.ac.kr)

However, datasets for sign language remain limited. There are various sign languages such as English, Korean, and Japanese. Many languages lack sufficient datasets, which degrades the performance of deep learning models. In general translation systems, when a language is input, the results are provided in sentence units rather than word units. Similarly, for sign language recognition systems to be useful in real life, they must go beyond simple word recognition and present results to non-deaf individuals in natural sentence form. Therefore, this study proposes a long short-term memory (LSTM)-based Korean sign language recognition and sentence conversion system that uses video datasets. The proposed system directly collects and automates datasets to address the problem of insufficient Korean sign language data. Moreover, the output of the results in word units can be difficult for users to understand as complete sentences. Thus, to recognize multiple words and convert them into sentence units, we sought to develop a practical system that can be utilized in real communication situations.

To recognize sign language, hand gestures and movements must be recognized in videos. In this study, Open Source Computer Vision (OpenCV) and MediaPipe were used for hand recognition. OpenCV is a library for real-time image processing and MediaPipe is a Google framework that offers various functions and models primarily for human body recognition. The proposed method uses features obtained from MediaPipe to train sign language gestures using LSTM. Among the deep learning models for sequential data, LSTM is used to learn continuous sign language video data. The LSTM has memory cells that can remember previous information for a long time, allowing it to learn long-term dependencies.

The remainder of this paper is organized as follows. Section 2 describes related research on sign language recognition and the models applied in this study. Section 3 explains the proposed LSTM-based Korean sign language recognition system. Section 4 verifies the performance of the proposed method and describes the results. Finally, Section 5 presents conclusions and future research directions.

2. Related Works

2.1 Research on Sign Language Recognition

With the development of computer vision and natural language processing, sign language recognition research has become an important issue. Sign language recognition research includes not only communication tools for the hearing-impaired but also technologies such as human-computer interaction and automatic interpretation. This study focuses on developing a system that recognizes sign language and performs sentence formation using computer vision and deep learning techniques.

To recognize sign language, it is necessary to identify hand shapes and movements from videos. Various studies have been conducted for this purpose. Chong and Lee [3] developed a sign language recognition prototype using a leap motion controller (LMC). While many previous studies have proposed incomplete sign language recognition methods, this study aimed at full American Sign Language (ASL) recognition, consisting of 26 letters and 10 numbers. Although most ASL letters are static, certain letters are dynamic. Therefore, this study extracted features from finger and hand movements to distinguish between static and dynamic gestures. The LMC was used for data collection in the sensor module, which included the 3-axis palm position, hand curvature radius, and positions of the five fingertips. All the collected data were processed using a preprocessing module to extract 23 meaningful features. The processing module provided classification results within a range of 1–36 using the support vector machine (SVM) and deep neural network (DNN). Gupta and Kumar [4] proposed an electronic wearable sign language recognition system aimed at developing a wearable translator for Indian Sign Language.

The system recognizes sign language by processing the data obtained using multiple surface electromyography (sEMG) sensors and inertial measurement units (IMUs) placed on both arms. Sign languages were categorized based on their lexical properties, and a multi-label classification was proposed to classify the signs. In addition, methods such as using accelerometers, gyroscopes, IMUs, sEMG sensors, Leap Motion, Kinect, and data gloves can be used to collect data on hand position, movement, and speed [5-7]. These sensor-based sign language recognition studies allow for accurate recognition of human movements based on sensor data. However, wearable or additional tools are required, and there may be spatial limitations.

Most systems that use cameras and deep learning have demonstrated sufficiently high recognition accuracy. In video-based sign language recognition, cameras are used to capture the hand movements and facial expressions of sign language users, which are then analyzed to recognize signs. Kim et al. [8] proposed a model that classified Alphabet finger spelling, a form of Alphabet sign language, using convolutional neural networks (CNN). This model recognizes patterns in learned finger-spelling signs and classifies the sign that corresponds to the observed action. It improves classification accuracy by utilizing not only hand landmarks but also hand regions. The dataset also consists of 24 classes, excluding the 26 letters of the alphabet, "J" and "Z" because they represent continuous behaviors. Lim et al. [9] proposed a model based on a CNN and hand energy image (HEI) to recognize single words or phrases. A CNN was used to model the hand shapes, whereas the HEI represented hand movements and positions. They used HEI as the input for the CNN to improve the accuracy of sign language recognition. Various studies on sign language recognition using deep learning are ongoing [10-12]. However, most studies have used a limited number of words, and research utilizing Korean sign language datasets remains insufficient. The studies mentioned earlier focus on training models with alphabetic sign language gestures or limited datasets, such as 11 signs in [10], 77 signs in [11], and 100 signs in [12]. Therefore, we automated the collection of Korean Sign Language data and built a dataset to enable the learning of various words.

2.2 LSTM

One of the main limitations of classical neural networks is their inability to consider temporal context. This means that they struggle to predict future events based on past information or to model continuous thought processes. To address this issue, a recurrent neural network (RNN) model was developed [13]. RNNs can capture temporal dependencies by passing information from the previous steps to the current step through a recurrent structure. As a result, RNNs are used in various fields, such as speech recognition, language modeling, translation, and image captioning [14,15]. However, RNNs face difficulties in capturing long-term dependencies as the input sequence lengthens. This is due to the gradual loss of the initial input information during processing. To solve this long-term dependency problem, LSTM was proposed. LSTM was developed in 1997 by Hochreiter and Schmidhuber [16] and was designed to selectively remember and utilize important information, even in long sequences.

A key feature of LSTM is the internal state called "cell state," which functions as a pathway for storing information over extended periods through the network. The cell state undergoes modifications and is propagated at each step of the LSTM and regulated to maintain the necessary information while discarding unnecessary data. The input gate determines the manner in which new information is added to a cell. This gate combines the current input and previous output to determine which information should be added to the cell state. The forget gate determines the information to be removed from the cell state. When this gate is activated, some information in the cell state is removed, which helps the model forget

unnecessary or outdated information. The output gate determines the portion of the cell state to be transmitted to the next layer. This gate combines the current cell state and input to select information for the output. Each LSTM cell receives input data and combines them with the cell state transmitted from the previous cell. The cell updates its state through the input, forget, and output gates, retaining only the essential information. This architecture enables the LSTM to effectively model long-term data dependencies and learn complex temporally varying data patterns.

Static sign-language images or individual character representations are primarily recognized using CNN models. In contrast, sign language words with dynamic characteristics are better suited for LSTM, which can consider the temporal context. LSTM specializes in processing sequential information, allowing it to effectively learn the temporal continuity of sign language gestures. Therefore, in this study, sequence data, in which the order of movements is meaningful, were used to train the LSTM.

3. Proposed Device Discovery Scheme

In this paper, we propose a Korean sign language recognition system to assist in communicating with individuals who are deaf or hard of hearing. The proposed system recognizes sign language gestures performed by a user in front of a camera in real time and converts them into text. For sign language recognition, the proposed method uses LSTM, which combines recognized words to generate natural sentences that are displayed in the user interface to facilitate smooth communication. Section 3.1 describes the process of this system, Section 3.2 explains the feature extraction and modeling processes from the video, and Section 3.3 outlines the automated data collection process for constructing the Korean sign language dataset.

3.1 Sign Language Recognition Process

This section provides a detailed explanation of the proposed method's mechanism and application scenarios. Fig. 1 is a diagram of the system process of the proposed method.

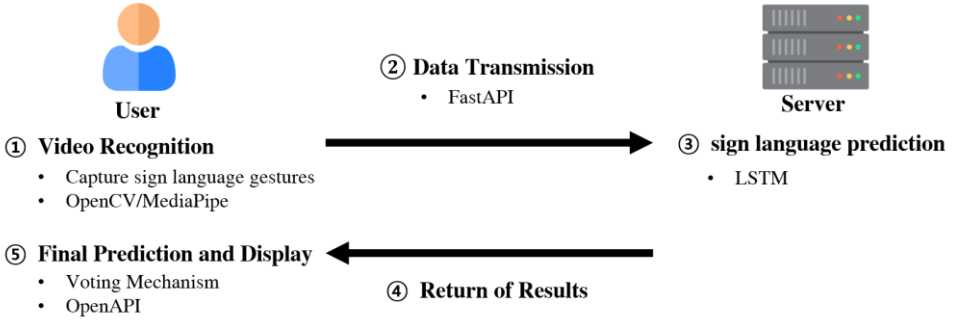

Fig. 1. Overview of the proposed system architecture.

The structure and operation of the proposed system are as follows. First, sign language gestures were captured in real-time using a camera in a user environment. The captured video was divided into frames and features were extracted from each frame. Instead of making direct predictions on the user's device, the extracted feature data were transmitted to a central server to improve network efficiency. In this

process, FastAPI was used to ensure fast and stable data transmission. The server then used the received feature data to predict the sign language via the LSTM. The prediction results obtained on the server were sent back to the user environment. In the user environment, a condition was set to detect the end of a word gesture in which the system required that the hands were not recognized for 10 consecutive frames. Once this condition was satisfied, a voting mechanism was applied to the accumulated prediction results to derive the final prediction. The final prediction result was displayed immediately through the user interface. Finally, once all the intended words had been recognized, an Open API was used to convert the recognized individual words into a grammatically natural complete sentence. Through this process, the system not only improved the accuracy of real-time sign language recognition but also ensured the efficient use of network and computational resources. In addition, natural language processing techniques were integrated to provide more flexible and comprehensible results, thereby enhancing the overall user experience.

3.2 Feature Extraction and Modeling Process

The proposed method utilized OpenCV and MediaPipe to extract hand and posture location information from camera footage. MediaPipe is an open-source framework that provides various body-recognition functions, allowing the vectorization of positional information between different body parts. MediaPipe Pose offers a feature for recognizing body posture, enabling vectorization of positional information between the shoulder and wrist. In this process, the coordinates of the shoulder and wrist are extracted, and the difference between the two coordinates is calculated and expressed as a vector. This vector represents the direction and distance from the shoulder to the wrist and can be used for posture analysis and motion recognition. In addition, MediaPipe Hands provides a function for recognizing hand shapes and vectorizing positional information between finger joints. After extracting the coordinates of each finger joint, the differences between the adjacent joints are calculated and expressed as vectors. These vectors represent the position and movement of each finger joint and can be used for hand motion recognition and gesture analysis. Furthermore, MediaPipe Hands provides detailed joint information for the thumb, index finger, middle finger, ring finger, and pinky, enabling precise handshape analysis.

MediaPipe's Hand and Pose components were used to analyze body postures and hand movements, and the angles of these postures and hand movements were calculated to store the data. First, the coordinates of the pose landmarks were stored in an array that included the x-, y-, and z-coordinates of each of the 33 landmarks. The shoulder and wrist coordinates were extracted to calculate the vectors between these two points. The shoulder is considered the parent joint and the wrist is considered the child joint; therefore, the vector between these joints was obtained. Subsequently, the calculated vector was normalized to create a unit vector with a length of 1. Normalization was performed by dividing each vector by its magnitude. The dot product of the vectors was computed to determine the angle between the normalized vectors, and the Arccos function was used to determine the angle. Finally, the pose landmark coordinates and calculated angles were combined into a single one-dimensional array to store the final pose data. This is the data storage process for each pose. The hand-gesture processes are similar. For poses, we extracted shoulder and wrist coordinates to calculate the vector between these two points and calculated the vector between the parent and child joints of the fingers. Using OpenCV and MediaPipe, we extracted 257 features per frame from a sign language video. To apply this to LSTM, we set 30 consecutive frames as one analysis unit and constructed an array of [30,257].

However, the following issues remain when training on individual words: sign language video clips are not of a consistent length. The padding technique is the most commonly used method to address this

problem. Owing to the data collection method, which adds training data through an API, it was difficult to determine the maximum sequence length, and there was a significant difference in the sequence length between short and long words. This can lead to reduced training efficiency and potential issues when receiving out-of-range inputs in real-service scenarios. Therefore, this study proposes a novel method to overcome the limitations of fixed-length input vector approaches and address the challenges associated with processing long input sequences. Traditional methods typically use a fixed-length sequence, such as 30 frames, padding shorter sequences with zeros to ensure uniformity. However, such approaches struggle to adapt to variations in gesture lengths and user motion speeds. To address these issues, the proposed method involves the following key steps. First, it dynamically segments input units based on the presence of hands in the frame using a dedicated algorithm. This approach ensures that the segmented units more accurately reflect the actual start and end of gestures. Second, while processing 30 frames as a single sequence, the model predicts multiple results for continuous gestures. A majority-vote mechanism is then applied to determine the final prediction, enhancing both accuracy and reliability. Finally, this method effectively accommodates variations in gesture lengths and user speeds, overcoming the inherent limitations of fixed-length input sequences. Subsequently, if no recognition occurred for more than ten frames, the system considered the word gesture to have ended and returned the result to the user's environment. To convert the words into sentences, the recognized words were sent to the GPT API once the user's input was confirmed to be complete. Using the GPT API, high-quality prediction and sentence completion functionalities can be provided to users.

3.3 Automated Data Collection

In this study, video data was collected using the API of the "Culture Open Data Portal" [17]. By requesting parameters, such as the service key, number of rows, and page number, the returned response includes the word name and video URL explaining the word in sign language. To manage the collected dataset, log files were recorded and saved as either txt or JSON files, depending on the purpose. The txt files were created to log the program operations and errors. Specifically, the txt files record requests for the sign language video data API, storage records of data conversions, file location changes/movements, and other tasks. JSON files were created to record information about the dataset. Before being processed into English, the original words and URLs of the original videos were stored in JSON files. This will allow future testing and various applications. These processes are illustrated in Figs. 2 and 3.

When storing data, two types of files are used: the original collected data file (RAW.npn) and the processed file (SEQ.npn) for model training. The RAW.npn file stores the video data vectorized on a frame by frame basis. The SEQ.npn file was converted into sequence data for model training, saved by truncating to a fixed length, and then overlapping. Although the SEQ.npn file can be loaded and used directly for training, it has the disadvantage of increasing size. As the volume of the collected dataset increases, the size of the SEQ.npn file also increases, leading to insufficient storage space on the computer. To address this issue, we improved the storage method as follows.

The RAW.npn files were retained, but the SEQ.npn files were deleted to free up storage space. However, this approach has the disadvantage of requiring sequences to be generated from scratch in the code each time the model training is conducted, which can be time consuming. To overcome this issue, the words were sequenced, grouped, and saved as compressed files (.npz). This allows for model training without the need to sequence each time; that is, simply select the desired group, decompress it, and proceed with training. Using this method, the storage size was reduced from 147 GB to 3.7 GB for 1,645 words. This process is illustrated in Fig. 4.

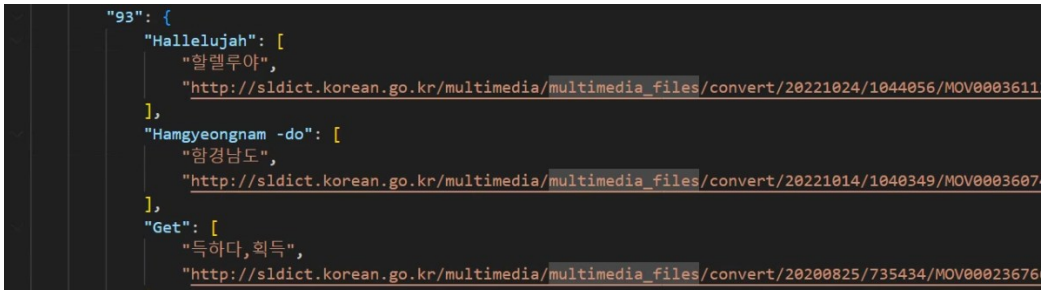

Fig. 2. txt log screen.

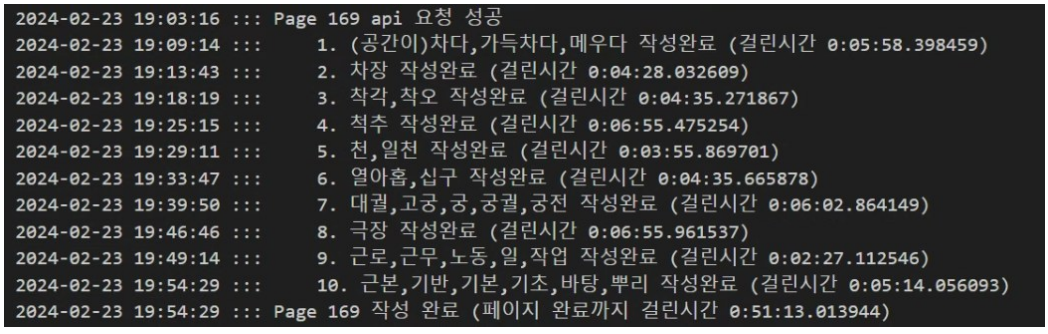

Fig. 3. JSON log screen.

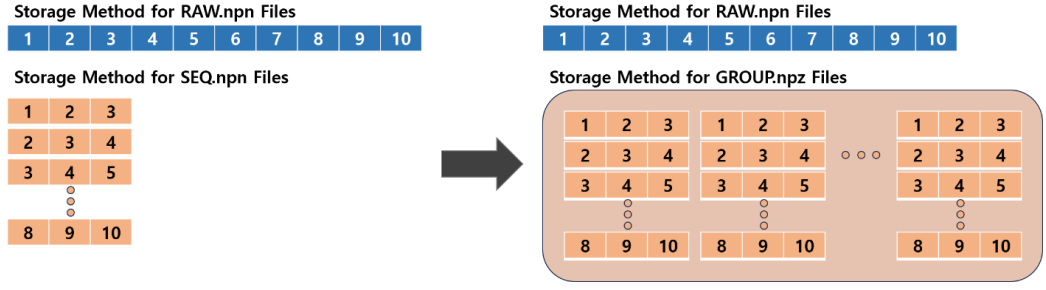

Fig. 4. Improved data storage method.

4. Experiment and Results

4.1 Datasets and Experimental Environments

For this experiment, the dataset used in this paper was collected using the API of the Culture Open Data Portal. The videos provided by the Culture Open Data Portal typically include only one video per word. To address the issues of data overfitting and insufficiency, four types of data augmentation were performed: [noise addition, vector quantization, time shift, and time delay]. Noise addition involved generating Gaussian noise and selectively adding it to specific features of the input data. This approach increased the variability of the dataset, enabling the model to learn effectively under diverse input conditions, preventing overfitting, and improving generalization performance. Vector quantization was applied by converting all values in the dataset into integers, creating integer-based features. This

simplification of data representation enhanced the stability of the learning process. Time delay was implemented by lengthening the sequence with a random warp factor, generating data with diverse temporal characteristics and increasing the diversity of the training dataset. Time shift involved randomly shifting the sequences by a certain amount and filling the empty spaces at the beginning or end with zeros, allowing the model to learn without relying on specific temporal alignments. By applying these data augmentation techniques, the diversity of the dataset was significantly increased, and the issue of overfitting was mitigated. The original dataset contained only one movement instance per word, which posed challenges such as the model's inability to recognize similar words or movements not included in the training data. Data augmentation addressed these issues and enabled the model to effectively learn under various input conditions. Data augmentation was applied to generate arrays from 3,552 words, which were subsequently divided into training and test sets in an 8:2 ratio. A summary of the dataset is provided in Table 1.

The experiment was conducted in the following hardware environment: The CPU was an Intel Core i9-10980XE with a clock speed of 3.00 GHz, and a system memory (RAM) of 256 GB. Four NVIDIA GeForce RTX 3090 GPUs were used as the graphics processing units (GPUs).

Table 1. Summary of the dataset

Dataset component	Value
Number of sequences	6,129,258
Sequence length	30 frames
Features per frame	257 features

4.2 Results of the Experiments

In this study, three model configurations were compared to select the one with the best performance. The configuration of each model is shown in Fig. 5. The proposed model architectures were designed to balance model complexity, learning capacity, and overfitting prevention. First, the LSTM layers were configured with 128, 256, and 1024 units to explore the impact of increasing model capacity on performance. In particular, larger unit sizes were intended to examine whether they could effectively learn complex sequential patterns in the data. Additionally, a dropout rate of 0.2 was consistently applied across all architectures to prevent overfitting, aiming to reduce the risk of overfitting while maintaining the model's expressiveness. All models used the Adam optimizer and sparse categorical cross-entropy loss function, and their accuracy was measured using the accuracy metric. The models were trained for up to 30 epochs and their performances were validated using a test dataset split at an 8:2 ratio.

The accuracy and loss values of each model are listed in Table 2. In this study, three LSTM-based model architectures for sign language recognition were designed and experimentally compared. The structure and performance of each model are as follows: Model 1 had the simplest architecture, consisting of a single LSTM layer (128 units), a dropout layer, and a dense output layer. This model achieved an accuracy of 93.76% with a loss of 0.3059. Model 2 featured a more complex structure, consisting of two LSTM layers (256 units each), two dropout layers, and two dense layers (a 128-unit intermediate layer and an output layer). This model achieved the highest accuracy of 96.08% and lowest loss of 0.2155 among the three models. Model 3 was designed using a single LSTM layer (1024 units), two dense layers (512 units and an output layer), and two dropout layers. This model exhibited an accuracy of 96.40% with a loss of 0.2631.

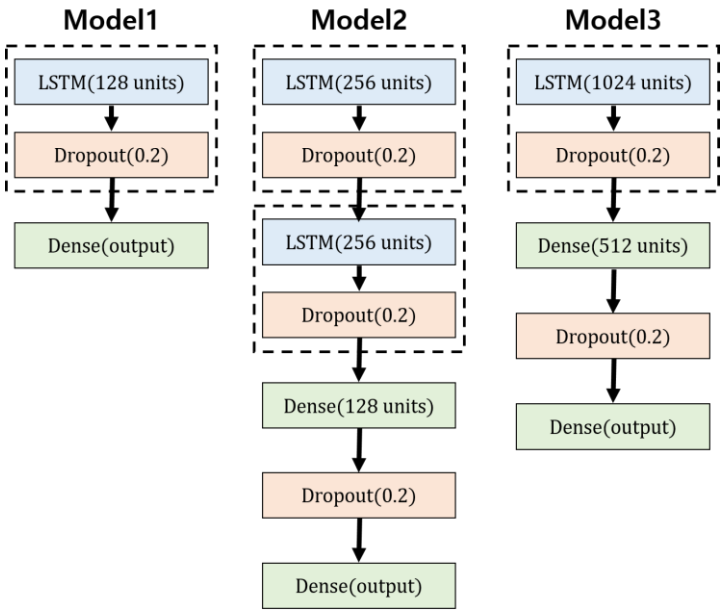

Fig. 5. Architecture of the model used in the experiment.

Table 2. Experimental results by model

	Accuracy (%)	Loss
Model 1	93.76	0.3059
Model 2	96.08	0.2155
Model 3	96.40	0.2631

The experimental results showed that Model 2 demonstrated the best performance, indicating that a deeper network structure with two LSTM layers could better capture the temporal characteristics of sign language gestures. Although Model 3 used only a single LSTM layer, it significantly outperformed Model 1, which can be attributed to the greater number of units (1024) and the addition of dense layers, which greatly enhanced the expressiveness of the model. These findings suggest that an LSTM-based model with an appropriate depth and complexity is effective for sign language recognition tasks. Specifically, the structure of Model 2, which combined multiple LSTM layers with proper dropout, proved capable of achieving high recognition accuracy while effectively preventing overfitting.

Figs. 6 and 7 show the interface screens of the proposed method. The interface of the developed sign language recognition system allows users to utilize various functions efficiently. As shown in Fig. 6, the user can activate the camera by entering “0” or viewing a video using the video URL. Through the “Turn to Sentence” function, recognized words can be converted into natural language sentences. Additionally, the “Clear Record” feature allows users to reset the record of recognized words and sentences. The "Model Load" function enables users to select and load the desired model to test various models. Finally, the “Set Sequence Length” feature allows users to modify the sequence length to fit the model, improving recognition accuracy. Fig. 7 shows the process of recognizing sign language videos. It displays the landmarks of the currently recognized hands and poses. Recognized words are shown at the bottom. This interface allows sign language recognition tasks to be performed intuitively and effectively.

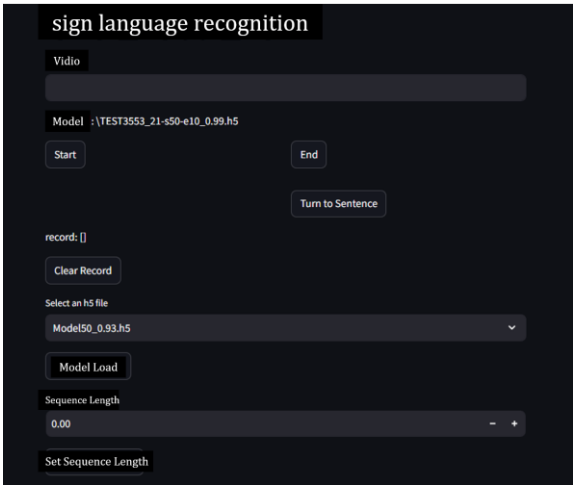

Fig. 6. Main screen of sign language recognition.

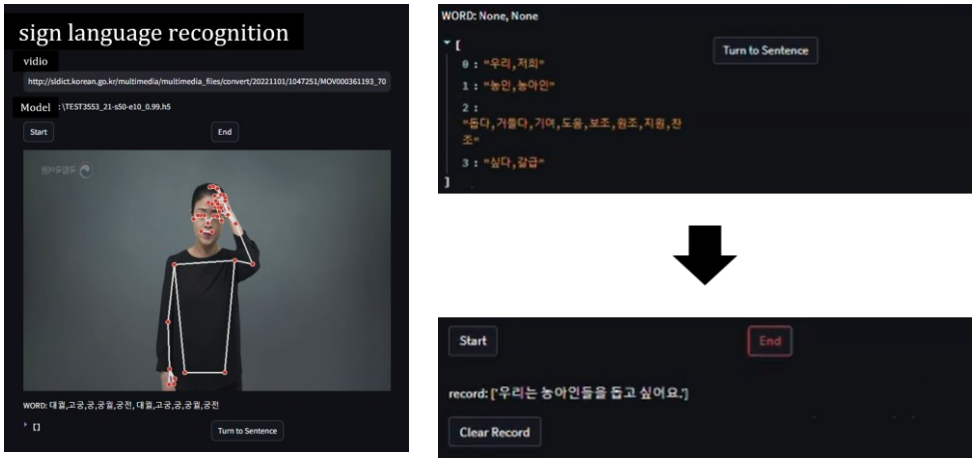

Fig. 7. (a) Screen of sign language recognition and (b) sentence conversion results.

5. Conclusion and Future Works

In this study, we propose an LSTM-based Korean sign language recognition system that uses video datasets to facilitate smooth communication with people with impaired hearing. To overcome the limitations caused by the lack of existing Korean sign language datasets, we collected and preprocessed sign language videos to construct our dataset. This allowed us to train various sign-language gestures and improve the recognition performance of the model. In addition, the proposed system utilized OpenCV and MediaPipe to extract hand movement and posture information from real-time videos, which were then input into LSTM for sign language recognition. Notably, instead of merely recognizing individual words, the system output results in sentence form, thereby enhancing user comprehension. The experimental results showed that the proposed system demonstrated a high level of accuracy in sign language recognition. Additionally, the user interface was designed to be intuitive and simple, allowing anyone to use it easily. This system is expected to contribute to improving communication among

hearing-impaired individuals and bridge the information gap.

Future research will require expanding the dataset to include more diverse sign language patterns and scenarios, as well as improving model accuracy. Additionally, we plan to evaluate whether the proposed method can operate efficiently in real-world application environments by measuring processing times across various conditions and conducting a quantitative comparative analysis. We also plan to introduce additional approaches to address the class imbalance issue in the dataset. By utilizing data augmentation techniques and class weighting, we aim to balance the model training and verify the resulting performance improvements. This will help further enhance the performance and reliability of the proposed method and lead to more generalized results.

Conflict of Interest

The authors declare that they have no competing interests.

Funding

This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP)-Innovative Human Resource Development for Local Intellectualization program grant funded by the Korea government (MSIT) (No. IITP-2025- RS-2022-00156334, 50%) and the National Research Foundation of Korea grant funded by the Korea government (MSIT) (No. 2021R1C1C2011105, 50%).

References

[1] J. W. Lee, B. M. Oh, J. H. Cho, and Y. S. Kang, “A study on the development process of sign language interpreting content in the medical setting,” *The Journal of the Korea Contents Association*, vol. 21 no. 12, pp. 505-516. 2021. https://doi.org/10.5392/JKCA.2021.21.12.505

[2] B. Garcia, and S. A. Viesca, “Real-time American sign language recognition with convolutional neural networks,” 2016 [Online]. Available: https://cs231n.stanford.edu/reports/2016/pdfs/214_Report.pdf.

[3] T. W. Chong and B. G. Lee, “American sign language recognition using leap motion controller with machine learning approach,” *Sensors*, vol. 18, no. 10, article no. 3554, 2018. https://doi.org/10.3390/s18103554

[4] R. Gupta and A. Kumar, “Indian sign language recognition using wearable sensors and multi-label classification,” *Computers & Electrical Engineering*, vol. 90, article no. 106898, 2021. https://doi.org/10.1016/j.compeleceng.2020.106898

[5] A. B. H. Amor, O. E. Ghoul, and M. Jemni, “Sign language recognition using the electromyographic signal: a systematic literature review,” *Sensors*, vol. 23, no. 19, article no. 8343, 2023. https://doi.org/10.3390/s23198343

[6] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, “Real-time human pose recognition in parts from single depth images,” *Communications of the ACM*, vol. 56, no. 1, pp. 116-124, 2013. https://doi.org/10.1145/2398356.2398381

[7] Y. Gu, C. Zheng, M. Todoh, and F. Zha, “American sign language translation using wearable inertial and electromyography sensors for tracking hand movements and facial expressions,” *Frontiers in Neuroscience*, vol. 16, article no. 962141, 2022. https://doi.org/10.3389/fnins.2022.962141

[8] G. Kim, S. Lee, C. Yoon, and S. Hong, “Design of finger sign language classification using convolutional neural networks,” *The Transactions of the Korean Institute of Electrical Engineers*, vol. 71, no. 10, pp. 1405-1410, 2022. http://doi.org/10.5370/KIEE.2022.71.10.1405

[9] K. M. Lim, A. W. C. Tan, C. P. Lee, and S. C. Tan, “Isolated sign language recognition using convolutional

neural network hand modelling and hand energy image," *Multimedia Tools and Applications*, vol. 78, pp. 19917-19944, 2019. https://doi.org/10.1007/s11042-019-7263-7

[10] D. Kothadiya, C. Bhatt, K. Sapariya, K. Patel, A. B. Gil-Gonzalez, and J. M. Corchado, "Deepsign: sign language detection and recognition using deep learning," *Electronics*, vol. 11, no. 11, article no. 1780, 2022. https://doi.org/10.3390/electronics11111780

[11] J. Shin, A. S. Musa Miah, M. A. M. Hasan, K. Hirooka, K. Suzuki, H. S. Lee, and S. W. Jang, "Korean sign language recognition using transformer-based deep neural network," *Applied Sciences*, vol. 13, no. 5, article no. 3029, 2023. https://doi.org/10.3390/app13053029

[12] R. Rastgoo, K. Kiani, and S. Escalera, "Video-based isolated hand sign language recognition using a deep cascaded model," *Multimedia Tools and Applications*, vol. 79, no. 31, pp. 22965-22987, 2020. https://doi.org/10.1007/s11042-020-09048-5

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533-536, 1986. https://doi.org/10.1038/323533a0

[14] J. Wang, Y. Yang, J. Mao, Z. Huang, C, Huang, and W. Xu, "CNN-RNN: a unified framework for multi-label image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 2285-2294. https://doi.org/10.1109/CVPR.2016.251

[15] A. Graves, A. R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proceedings of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, Canada, 2013, pp. 6645-6649. https://doi.org/10.1109/ICASSP.2013.6638947

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, 1997. https://doi.org/10.1162/neco.1997.9.8.1735

[17] Culture Open Data Portal [Online]. Available: https://www.culture.go.kr.

Soo-Yeon Jeong **https://orcid.org/0000-0001-8118-7054**

She received B.S. degree in computer engineering from Daejeon University in 2015, M.S. degree in computer engineering from Chungnam National University in 2017, and Ph.D. degree in computer engineering from Chungnam National University in 2022. She is currently an assistant professor in the Division of Software Engineering at Pai Chai University, Daejeon, Korea. Her research interests include recommender systems, context awareness, and data mining.

Ho-Yeon Jeong **https://orcid.org/0009-0001-2873-9849**

He received B.S. degree in Drone and Robot Engineering from Pai Chai University in 2025. He is currently a master's student in the Department of Artificial Intelligence at Kyung Hee University. His research interests include deep learning and computer vision.

Sun-Young Ihm **https://orcid.org/0000-0002-7545-7035**

She received B.S. degree in multimedia science from Sookmyung Women's University in 2011, M.S. degree in multimedia science from Sookmyung Women's University in 2013, and Ph.D. degree in IT engineering from Sookmyung Women's University in 2017. From 2017 to 2019, she was a senior researcher at the Big Data Utilization Research Center, Sookmyung Women's University. From 2018 to 2019, she was an invited professor in the Department of IT Engineering at Sookmyung Women's University. From 2020 to 2021, she was a research professor at the CRC Research Center, Dongguk University. She is currently an assistant professor in the Department of Computer Engineering at Pai Chai University, Daejeon, Korea. Her research interests include databases, top-k query processing, big data, machine learning, and deep learning.