processes

Article

Application of Wearable Gloves for Assisted Learning of Sign Language Using Artificial Neural Networks

Hyeon-Jun Kim [1] and Soo-Whang Baek [2,*]

[1] Department of Electronic Information System Engineering, Sangmyung University, Cheonan 31066, Republic of Korea
[2] Department of Human Intelligence and Robot Engineering, Sangmyung University, Cheonan 31066, Republic of Korea
* Correspondence: swbaek@smu.ac.kr; Tel.: +82-41-550-5543

Abstract: This study proposes the design and application of wearable gloves that can recognize sign language expressions from input images via long short-term memory (LSTM) network models and can learn sign language through finger movement generation and vibration motor feedback. It is difficult for nondisabled people who do not know sign language to express sign language accurately. Therefore, we suggest the use of wearable gloves for sign language education to help nondisabled people learn and accurately express sign language. The wearable glove consists of a direct current motor, a link (finger exoskeleton) that can generate finger movements, and a flexible sensor that recognizes the degree of finger bending. When the coordinates of the hand move in the input image, the sign language motion is fed back through the vibration motor attached to the wrist. The proposed wearable glove can learn 20 Korean sign language words, and the data used for learning are configured to represent the joint coordinates and joint angles of both the hands and body for these 20 sign language words. Prototypes were produced based on the design, and it was confirmed that the angle of each finger could be adjusted. Through experiments, a sign language recognition model was selected, and the validity of the proposed method was confirmed by comparing the generated learning results with the data sequence. Finally, we compared and verified the accuracy and learning loss using a recurrent neural network and confirmed that the test results of the LSTM model showed an accuracy of 85%.

Keywords: artificial intelligence; internet of things; wearable; neural network; RNN; LSTM

Citation: Kim, H.-J.; Baek, S.-W. Application of Wearable Gloves for Assisted Learning of Sign Language Using Artificial Neural Networks. *Processes* **2023**, *11*, 1065. https://doi.org/10.3390/pr11041065

Academic Editors: Wen-Jer Chang and Jiangxin Wang

Received: 16 February 2023
Revised: 22 March 2023
Accepted: 31 March 2023
Published: 1 April 2023

1. Introduction

Language is a means of communication for humans that affects the development of social skills and the establishment of self-identity. Sign language has been used as a means of communication for the deaf and has its own language system [1]. Sign language is visual and conveys meanings through hand movements, arm positions, and mouth shapes [2]. Sign language establishes a sense of belonging and identity for people with deafness and plays an important role in communication and academic achievement [3]. However, because sign language is a visual language, it is difficult for nondisabled people who do not know it to understand its meaning. In addition, it is difficult for nondisabled people without hearing impairments to accurately convey the meaning of sign language expressions unless they are sign language experts. Some teachers who teach people with deafness use spoken language and poor sign language expressions to teach classes; for this reason, people with deafness sometimes do not clearly understand the content being taught. Thus, a phenomenon has been observed in which students study according to the will of their parents, peers, and individuals rather than the will of their teachers [4]. When people with deafness go to normal schools, the services provided usually include sign language interpreters. However, if you become accustomed to a sign language interpreter, you may have difficulty learning your native language [5]. In this way, communication problems arise between people with deafness who use sign language and nondisabled people who use spoken language. For this reason, research is being conducted to understand and classify the sign language expressed by the deaf [6].

Research on sign language recognition is being conducted in two main ways. The first method uses a sensor to recognize sign language. One frequently used method of using sensors is inertial measurement units (IMUs). For example, one study used an IMU and a surface electromyography (sEMG) sensor. In this study, arm gestures were identified using IMUs, hand gestures, which were discriminated by applying the sEMG sensor, and a wearable system that recognized American sign language (ASL) in real time. This was implemented by fusing information from IMU and sEMG sensors [7]. In addition to this, a study on sign language translation used an IMU and a flex sensor. This study developed an Android-based mobile application with a text-to-speech function that recognizes the alphabet, transmits it to a mobile device through Bluetooth communication, and listens to the received text [8]. Another study used a touch sensor to recognize A–Z and 0–9 in ASL. In this study, a device for recognizing hand gestures was implemented using eight touch sensors. A charge-transfer touch-sensor-based gesture recognition glove for ASL translation was presented, with the advantage of being portable and being possible to implement using inexpensive hardware [9]. There was also a study on sign language recognition using a yarn-based stretchable sensor array (YSSA). A highly flexible sensor was used to recognize sign language and convert the recognized sign language into voice, and it was shown that a wearable sign language translation system supporting machine learning can accurately translate ASL hand gestures into voice [10]. One study on sign language recognition used inertial sensors and magnetometers. A sign language recognition system using a data glove consisting of a three-axis accelerometer, magnetometer, and gyroscope was presented, and a study was conducted to output a three-dimensional sign language model on a personal computer (PC) monitor [11].

The second method is to recognize sign language using deep learning. Accordingly, sign language is recognized using deep learning models, such as the convolutional neural network (CNN), three-dimensional convolutional neural network (3DCNN), and LSTM [12–14]. Artificial intelligence has been the most active research area recently. Research on deep learning training using dynamic data is in progress, as is research using RNN-based deep learning training models according to the characteristics of data that change over time [15]. When training a training model, some studies train this model by combining two or more models instead of only one model. Some studies combined the CNN and the RNN to recognize sign language. This study improved the training performance of the training model by training spatial features with the CNN and temporal features with the RNN [16]. In addition, when using a single model, some studies improve its training performance by changing the structure of the model. These studies improve training performance by changing and combining the structure of LSTM and the gated recurrent unit (GRU) [17]. In addition, in the study of sign language recognition using artificial intelligence, one study improved sign language recognition through the transformation of training data. Another study improved performance by reinforcing the feature vector of each hand in the data-preprocessing process. It was suggested that it is possible to solve the error caused by overlapping hand motions [18]. In addition, one study reduced the size of the dataset by converting video data into image data and improved sign language recognition using these data [19].

There have also been studies that apply sign language recognition techniques to sign language education. Representative examples include studies using humanoids and animations. In studies using humanoids, the humanoids use sign language so that humans can understand them [20,21]. Animation-based sign language education, similar to humanoid education, expresses input words as animations so that students can observe and reproduce them [22]. This type of sign language education has the advantage of allowing for the user to learn sign language expressions at the place and time of their choosing. However, because it is a learning method that humans observe and follow, it has a disadvantage in that it is difficult to learn the exact hand motions or arm positions.

In order to eliminate these drawbacks, some studies have proposed wearable gloves that operate based on algorithms that use both visual and tactile feedback. The wearable glove is designed to enhance the effectiveness of education using haptic technology. When a physical stimulus is applied to the human skin, the brain recognizes this and activates the relevant muscles. Technologies that use this phenomenon are called haptic technologies. The term "haptics" is derived from the Greek word "haptikos," which means tactile sensation, and haptic technology is a technology that allows for users to feel tactile sensations by applying force, vibration, and movement [23]. When a stimulus is applied to the skin with haptic technology, the brain accepts this stimulus. Correspondingly, learning ability can be improved using the visual and tactile senses [24]. Forms of education using haptic technology are being studied in many fields, such as medicine, art, and sports. In the medical field, it is used in subfields, including virtual and dental surgeries [25,26]. In the art field, there are studies on the use of haptic technology in various subfields, including piano education [27]. In the sports field, numerous studies have been conducted, including on tennis exercises [28]. These studies indicate that user education can be improved through the use of haptics such as vibration.

The wearable glove proposed in this study provides haptic feedback through vibration and movement. To express sign language words, each finger is controlled by a direct current (DC) motor and a lead screw. When controlling each finger, the finger angle of the wearable glove is measured using a flex sensor. When the movement of the arm is recognized in the input image, the vibration motor vibrates in the direction in which the arm should move to provide information on the movement direction. Based on this design, a prototype was produced to confirm that it is possible to control each finger's angle. The deep learning model is used to recognize sign language, and the proposed wearable glove uses the LSTM model. Thus, data on the joint coordinates and joint angles of both hands and body can be utilized to learn 20 Korean sign language expressions. To express sign language symbols, both the angle of the finger and all visible parts, such as the position of the arm and the shape of the mouth, must be controlled. However, in this study, sign language expressions are limited as only the direction of movement of each finger and arm are controlled. To evaluate the performance of the sign language recognition model of the proposed wearable glove, an RNN and LSTM are trained, and accuracy and loss are measured and compared.

2. Sign Language Training

2.1. Sign Language Training Model

Figure 1 shows the sign language training architecture of the LSTM model. The wearable glove being used as a sign language training model was trained using LSTM. LSTM is a type of RNN. In existing RNNs, the current calculation result depends on the previous calculation result [29]. An RNN is more effective at learning using short sequences. If the RNN sequence is long, it is not possible to transfer sufficient information from the previous sequence to the subsequent sequence. LSTM compensates for these disadvantages of the RNN. LSTM preserves past computational results by leaving meaningful data intact and erasing meaningless data. As a result, even when learning with a long sequence, previous calculation results can be used, and learning can proceed. LSTM cells consist of six parameters and three gates. The six parameters are the input, output, parameter c_{t-1} of the previous cell state, parameter c_t of the current cell state, parameter h_{t-1} of the previous hidden state, and parameter h_{t-1} of the current hidden state. The input is a parameter to which data used for learning are input, and the output is a parameter to which results are output through all gates. The parameter input is used to input the data used for training, and the parameter output is used to output results after they pass through all the gates. In addition, c_{t-1} stores the calculated result in the previous cell and inputs it into the current cell, and c_t stores the calculated result in the current cell and inputs it into the next cell. The cell state has a structure that can maintain the calculated results for a long time. h_{t-1} is the parameter in which the value calculated by passing the gate in the previous cell is stored. At h_{t-1}, new input data x_t are combined and passed through the gate, and the calculated result and output are stored in h_t. The forget gate removes unnecessarily calculated results from the previous cell's calculated results. The input gate stores information about time t. This gate combines the newly entered data x_t and h_{t-1} to determine the required computational result and sends this to the cell state. The output gate determines the value to be output and calculates and outputs the values of parameters h_{t-1}, x_t, and the cell state. As sign language is a visual language that expresses meaning by changing the shape of the hand and position of the arm over time, the training was conducted with an LSTM model that is easy to train over time, and a selection of 20 words was used as the data, as summarized in Table 1. Considering the time required for learning, the total number of videos was set to 60. The video used for sign language training was 4 or 5 s long on average, and the video frame was 30 fps.

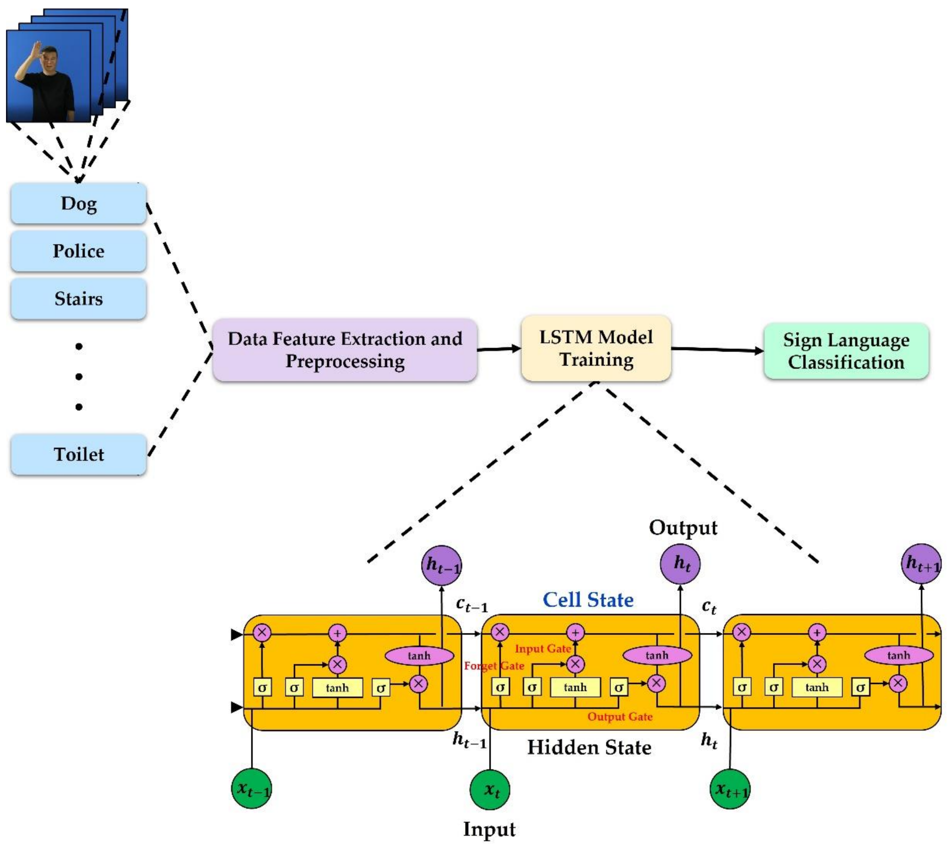


Figure 1. Sign language training architecture of LSTM model.

Table 1. Sign language words used in training.

Sign Language Word	Total Number of Videos	Time of Video (s)	Frames of Video (fps)
Dog	60	4	30
Police	60	4	30
Stairs	60	4	30
Moon	60	4	30
Bat	60	5	30
Bee	60	4	30
Hospital	60	5	30

Table 1. *Cont.*

Sign Language Word	Total Number of Videos	Time of Video (s)	Frames of Video (fps)
Bandage	60	5	30
Teacher	60	4	30
Baby	60	4	30
Apartment	60	4	30
Dizziness	60	4	30
Elevator	60	5	30
Glass	60	4	30
Food	60	4	30
Car	60	4	30
Toy	60	5	30
Thermometer	60	5	30
Friend	60	4	30
Toilet	60	5	30

2.2. Data Preprocessing

Data preprocessing is an essential process for deep learning training. This study used the MediaPipe Holistic Application Programming Interface (API) to transform data. The coordinates of each joint and fingertip were extracted using the MediaPipe Holistic API [30]. Figure 2 shows the hand and pose landmarks provided by the MediaPipe Holistic API. Twenty-one hand landmarks could be extracted, including the coordinates of each joint and fingertip of the recognized hand during hand recognition. Using the MediaPipe Holistic API, information for the left and right hands can be recognized separately. Thirty-three pose landmarks could be extracted, including those for the eyes, nose, mouth, and all joints of the body.

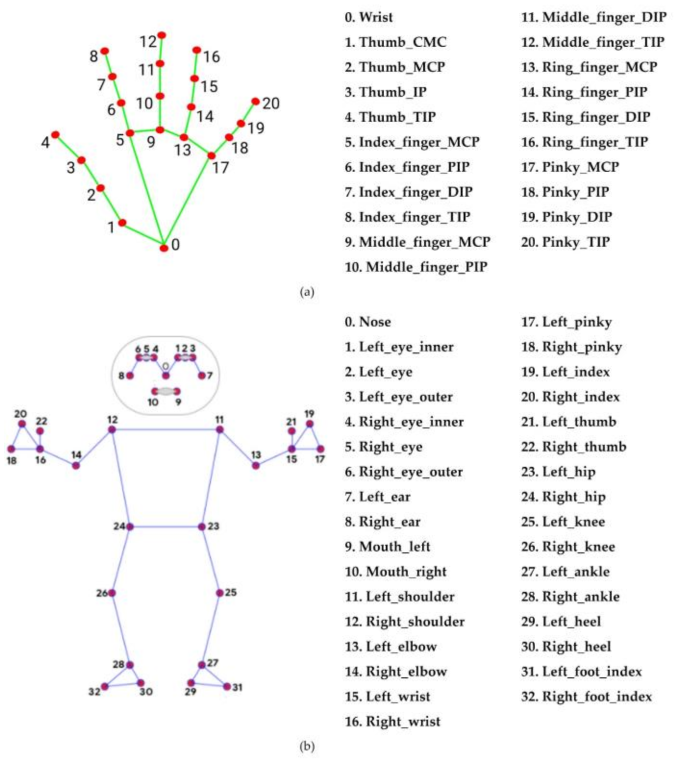


Figure 2. MediaPipe Holistic API [31,32] (**a**) the hand and (**b**) the body.

Three coordinates can be defined to extract the angles denoted as α, β, and γ, where α is the starting point of the first vector, β is the endpoint of the first vector and the starting point of the second vector, and γ is the endpoint of the second vector. Vectors $\vec{a}$ and $\vec{b}$

obtained through this method are expressed using Equations (1) and (2). The angle θ subtended by the two vectors can be expressed using Equation (3). Based on this, the joint angles can be extracted.

$$\vec{a} = Landmark[\alpha] - Landmark[\beta] \tag{1}$$

$$\vec{b} = Landmark[\beta] - Landmark[\gamma] \tag{2}$$

$$|\vec{a}| \times |\vec{b}| \times \cos\theta = \vec{a} \cdot \vec{b} \tag{3}$$

The MediaPipe Holistic API can extract data for the left and right hands. The right-hand information was extracted in the form of right_hand_landmarks, and the left-hand information was extracted as left_hand_landmarks. Recognized landmarks contain the coordinates x, y, and z, as well as visibility information. Figure 3 shows the landmarks of both hands and the skeleton of the hand generated using the MediaPipe Holistic API. Using the landmark's x and y coordinates, the corresponding landmark was assigned a number. When the right hand was recognized, the sentence "right hand" is displayed, and when the left hand is recognized, the sequence "left hand" was displayed to indicate which hand had been recognized.

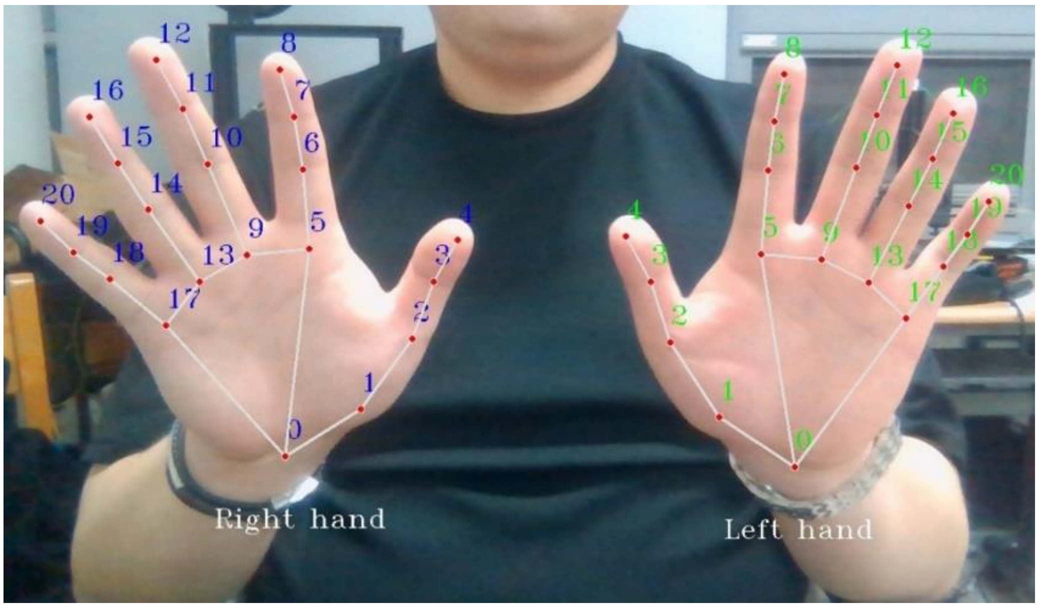


Figure 3. Two-handed recognition scheme using MediaPipe Holistic API hand landmarks.

Figure 4 shows the landmarks and skeleton of the body generated using the pose landmarks of the MediaPipe Holistic API. The MediaPipe Holistic API extracts body skeletal data as pose landmarks and can extract 33 landmarks in total. Each landmark was extracted, totaling 11 landmarks on the face, 12 landmarks on the upper body, and 10 landmarks on the lower body. Recognized landmarks had x, y, and z coordinates and similar visibility information to the hand landmarks. As shown in Figure 4, landmarks and skeletons can be recognized even if body parts are not recognized.

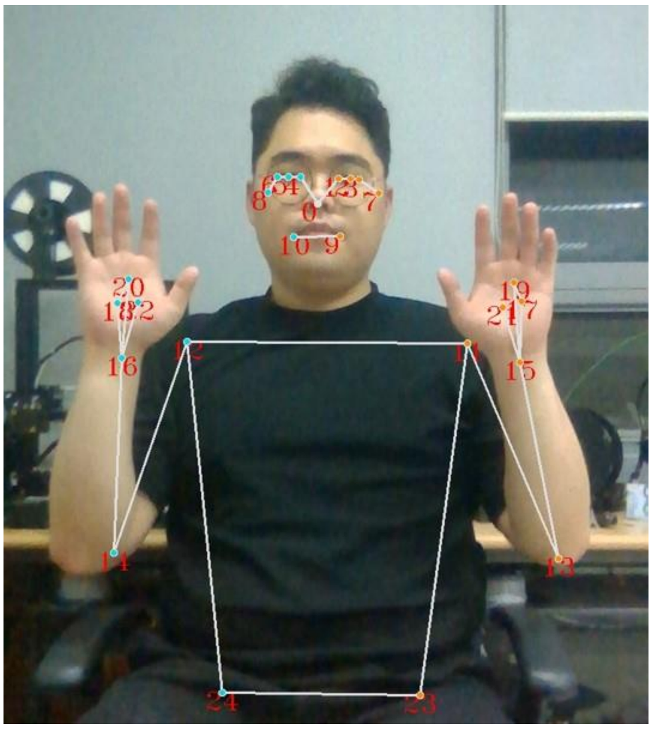


Figure 4. Skeletal recognition using MediaPipe Holistic API pose landmarks.

2.3. *Training Data*

The data used for the LSTM model were trained using the sign language dataset provided by the AI-Hub [33]. As listed in Table 1, the data were videos that corresponded to 20 Korean sign language words. For each word, 20 sign language experts performed the sign language movements and used videos filmed from the anterior, right lateral, and left lateral sides of each participant. Figure 5 displays the skeletal data of the body and each hand using the MediaPipe Holistic API for the sign language expression that corresponds to “car”. The data extracted for each hand consisted of information about the x, y, and z coordinates and the visibility of each landmark. The data extracted for the body also comprised information about the x, y, and z coordinates and the visibility of each landmark, similar to the data extracted for the hand. A total of 84 pieces of data were extracted for the hand, and a total of 132 pieces of data were extracted for the body.

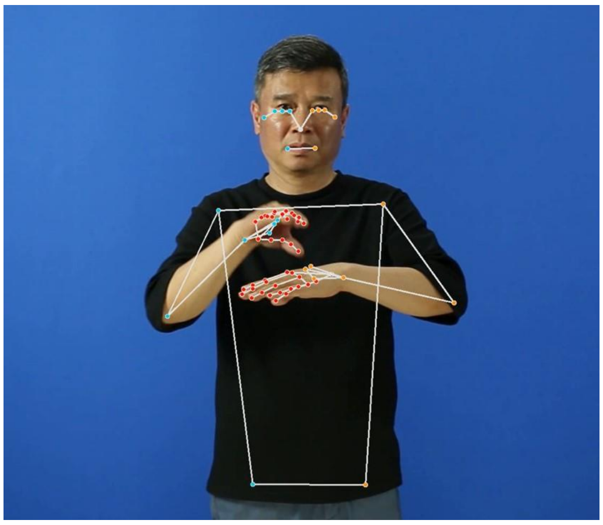
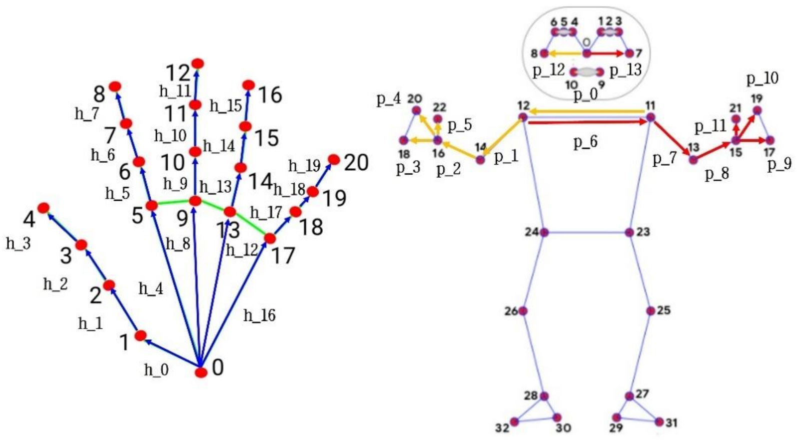


Figure 5. Video data used for learning (sign language expression for car).

Angle data for each hand and the body were obtained using Equations (1)–(3), as described in the previous section. The hand and pose landmarks used for angle data extraction are shown in Table 2, and the extracted vectors are shown in Figure 6. As shown in Table 2 and Figure 6, 20 vectors were extracted from each hand and 14 vectors were extracted from the body. The symbols h_0–h_3 represent vectors for the thumb. The vector from the wrist to the thumb's carpometacarpal (CMC) joint is h_0, the vector from the thumb's CMC joint to the thumb's metacarpophalangeal (MCP) joint is h_1, the vector from the thumb's MCP joint to the thumb's interphalangeal (IP) joint is h_2, and the vector from the thumb's IP joint to the thumb's tip is h_3. If this process is performed on all fingers, 20 vectors can be extracted in total. For the body vectors, the vectors from the nose to both ears and the upper body vector were also extracted. The symbols p_0–p_5 denote the vectors for the right arm, and the symbols p_6–p_11 denote the vectors for the left arm. The vector from the left shoulder to the right shoulder is p_0, the vector from the right shoulder to the elbow is p_1, the vector from the elbow to the wrist is p_2, and the vectors from the wrist to the pinky, index, and thumb are p_3, p_4, and p_5, respectively. The left arm also extracted vectors from p_6 to p_11 using the same process. The vectors corresponding to the right and left ears with respect to the nose are p_12 and p_13, respectively. All the extracted vectors were converted into unit vectors with equalized sizes. The angle between two unit vectors can be calculated as the dot product of the two vectors, as shown in Equation (3). The angles calculated for both hands were calculated with the vectors, from the wrist to the tips of each finger. Fifteen angles could be calculated by calculating three angles per finger. The angles to the body were calculated in the same way. Five angles could be calculated from one arm, in addition to the angles from the nose to both ears; thus, eleven angles could be calculated in total.

Table 2. Vectors extracted and landmarks used.

Hand Landmark	Vector Extracted (Hand)	Pose Landmark	Vector Extracted (Pose)
0, 1	h_0	11, 12	p_0
1, 2	h_1	12, 14	p_1
2, 3	h_2	14, 16	p_2
3, 4	h_3	16, 18	p_3
0, 5	h_4	16, 20	p_4
5, 6	h_5	16, 22	p_5
6, 7	h_6	12, 11	p_6
7, 8	h_7	11, 13	p_7
0, 9	h_8	13, 15	p_8
9, 10	h_9	15, 17	p_9
10, 11	h_10	15, 19	p_10
11, 12	h_11	15, 21	p_11
0, 13	h_12	0, 8	p_12
13, 14	h_13	0, 7	p_13
14, 15	h_14		
15, 16	h_15		
0, 16	h_16		
17, 18	h_17		
18, 19	h_18		
19, 20	h_19		



(a) (b)

Figure 6. Vectors extracted for (**a**) the hand and (**b**) the body.

Table 3 lists the landmark's x, y, and z coordinates, visibility data, angle data of each joint, and the total sum of the data. As indicated in the listings in Table 3, when the x, y, and z coordinates and the visibility data were extracted, 84 data segments were extracted. In total, 100 pieces of data were extracted by adding the angles of each finger and the correct answer labels. When the x, y, and z coordinates and visibility data were extracted, 132 pieces of data were extracted. A total of 144 data segments were extracted by adding the angles and correct answer labels. To generate data to be trained on the LSTM model, the sum of the two datasets must be equal to 100. Therefore, the data extracted from the body must be processed to match the sum of the data obtained from the hand. When using sign language expressions, information about the lower body is not included. For this reason, the data from pose landmarks 23–32, that is, the x, y, and z coordinates and visibility data of the landmarks corresponding from the hip to the foot, were eliminated; these data yielded 100 body datasets. Based on this process, the LSTM model training data were created by equalizing the sizes of the two datasets.

Table 3. Data preprocessing: the landmark's x, y, and z coordinates, visibility data, the angle data of each joint, and the sum of the data.

Index	Hand	Body
Landmark's x, y, and z coordinates and visibility information	84	132
Angle	15	11
Answer label	1	1
Deleted data	0	44
Total	100	100

3. Wearable Glove

Figure 7 shows the configuration of the wearable glove. As shown in Figure 7, when the video was input, sign language was recognized using the deep learning model, and the angle of each finger and the direction of the arm movement were input to the micro-controller unit (MCU). Accordingly, the DC motor attached to each finger controlled the finger according to the input angle, and the four vibration motors attached to the wrist informed the movement direction of the arm by vibrating in the direction the arm should move. The wearable glove is composed of control, driving, and input units. The control unit is composed of a personal computer that receives video input, recognizes the sign language motion of the input video, and transmits the corresponding sign language finger angle and arm movement direction to the MCU; the MCU controls the DC and vibration motors. Specifically, the MCU controls five DC motors, five flex sensors, and four vibration motors. It uses an Arduino Mega 2560 R3 board comprising 54 digital input/output pins and 16 analog input pins. The drive unit controls the exoskeleton of the wearable glove with five DC motors (RA-25GM) with an input voltage of 12 V and a rated output power of 22 W; this consists of a vibration motor (ELB060416) attached to the four sides of the wrist to indicate the direction of the arm movement. The input unit consists of the wearable glove and the flex sensor (SZH-SEN02) that inputs the finger angle according to the sign language expressed by the DC motor, which is communicated to the MCU.

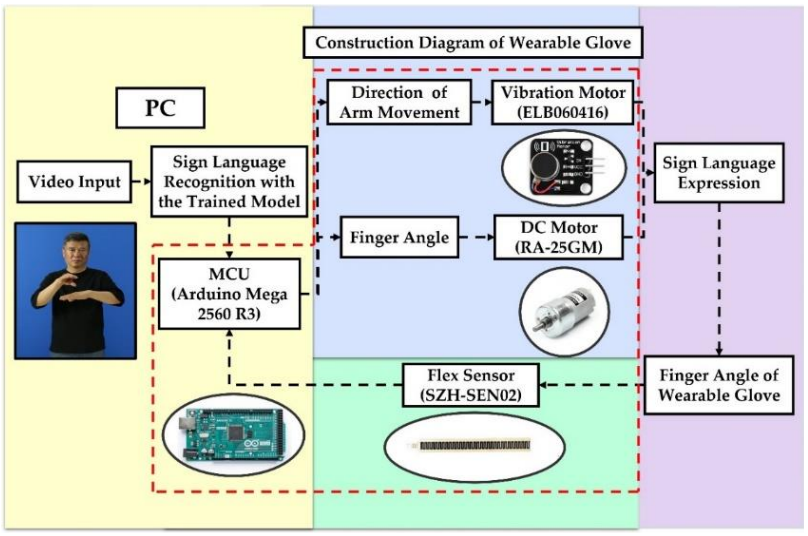

Construction Diagram of Wearable Glove
PC
Video Input
Sign Language Recognition with the Trained Model
Direction of Arm Movement
Vibration Motor (ELB060416)
Sign Language Expression
Finger Angle
DC Motor (RA-25GM)
MCU (Arduino Mega 2560 R3)
Flex Sensor (SZH-SEN02)
Finger Angle of Wearable Glove

Figure 7. Configuration diagram of the wearable glove.

Figure 8 is a hand model that represents the design of the wearable glove. The hand model was designed based on the actual wearer's hand. The width of the palm was 85 mm and the length was 93 mm. The thicknesses of the thumb, index finger, middle finger, ring finger, and pinky were 30.5 mm, 23 mm, 23.5 mm, 21 mm, and 20 mm, respectively. The lengths of the thumb, index finger, middle finger, ring finger, and pinky were 114.5 mm, 71.5 mm, 80.5 mm, 72.5 mm, and 54.5 mm, respectively. Based on this hand model, the wearable glove was designed by applying each dimension corresponding to the fingers and back of the hand.

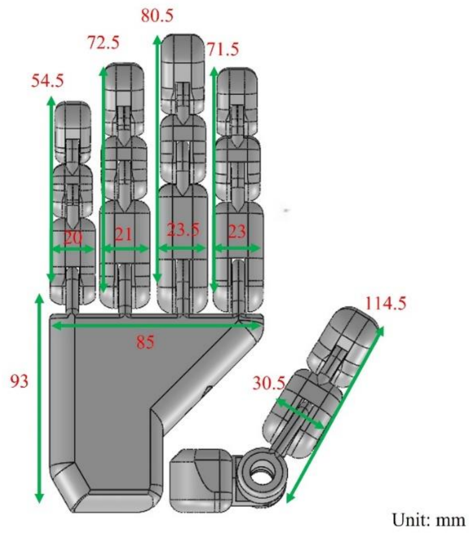


Figure 8. Hand dimensions.

Owing to their characteristics, exoskeleton wearable robots inevitably cause discomfort to users. To solve this problem, the wearable glove was designed to minimize the interference between the fingers. Figure 9 shows a model of the wearable glove. When the user puts on the wearable glove, the exoskeleton is only attached to the posterior side of the hand, thus achieving a comfortable fit. In addition, as the exoskeleton frame attached to each finger joint is fixed along the direction of the posterior part of the hand rather than wrapped around the finger, it is designed to minimize the interference among the fingers caused by the wearable glove. As shown in Figure 9, the wearable glove has one DC motor attached to each finger. The lead screw, with a length of 100 mm and a pitch of 8 mm, is fixed to each DC motor. The part connected to the lead screw is a frame used to connect the shaft of the wearable glove. This part is fixed to the motor frame of the wearable glove and enables linear motion such as that generated by a linear motor. As the nut also has the same pitch, it is possible to achieve linear displacements equal to 8 mm per degree of angular motor rotation. The motor was fixed to the thumb with an independent frame. As shown in Figure 9, as the motor attached to the other finger is fixed in the opposite direction, when controlling the wearable glove, the motor of the thumb is controlled by generating a signal opposite to the motor attached to the other finger. A separate flex sensor (used to measure the angle of the finger) is attached to each finger. The flex sensor used is a sensor with a length of 130 mm and a width of 15 mm. The flex sensor is fixed underneath the exoskeletal frame on each finger. In this way, the degree of bending of each finger can be measured.

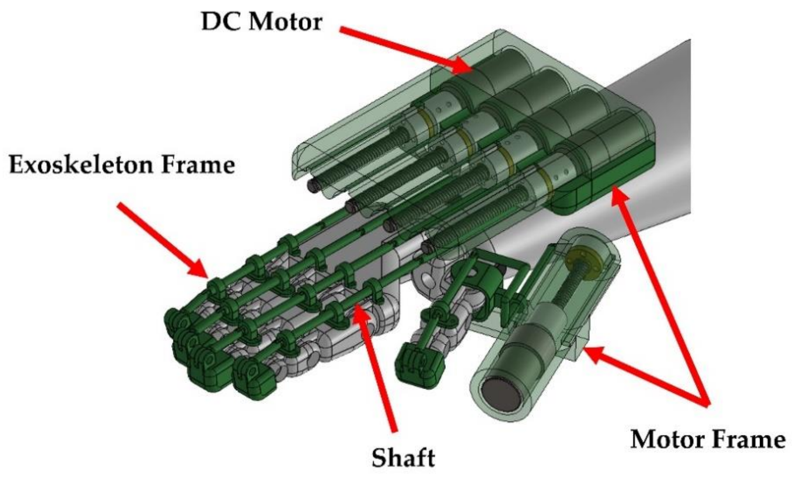


Figure 9. Overall model of the wearable glove.

Figure 10 shows the vibration motors attached to the wrist. The frame of the wearable glove is attached and fixed to the cotton glove. The vibration motor of the wrist part is independently fixed to the wearable glove. The vibration motors are fixed to the upper (number 1), lower (number 2), right (number 3), and left (number 4) sides of the wrist. In the sign language recognition step, the input video is recognized as a sequence of planar images in the x–y direction. At this time, the data from the skeleton of the sign language expert are extracted from the video. Using the recognized skeleton data, the x and y coordinates of the wrist are stored at time t, the movement of the arm is calculated by comparing the x and y coordinates of the wrist in the previous sequence as t − 1, and the difference is calculated. When y changes in the +y direction, vibration motor 1 moves, and when it changes in the −y direction, vibration motor 2 moves. When x changes in the −x direction, vibration motor 3 moves, and when it changes in the +x direction, vibration motor 4 moves. When both the x and y directions change, the movement of the arm is directed by all the vibration motors being driven in the corresponding direction.

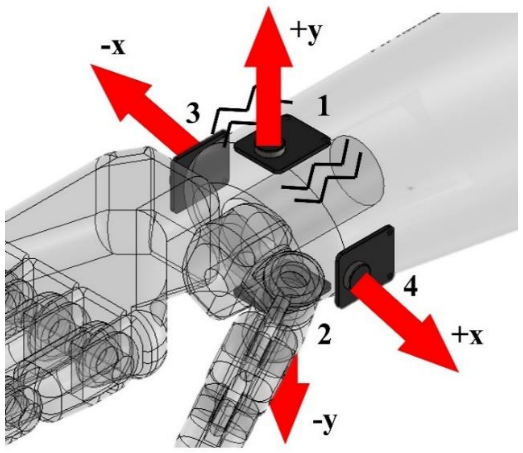


Figure 10. Vibration motors in the wearable glove.

3.1. Operational Algorithm of Wearable Glove

The operational algorithm of the wearable glove is represented in Figure 11. As shown, all the fingers are extended in the default state of the wearable glove. Thereafter, when the video is input to the PC, the trained model recognizes the sign language expression of the input video frame. When the sign language expression is recognized, the finger angle and arm movement direction are stored as an array at time t. After their storage, the input sign language video is output, and the wearable glove is moved. The DC motor connected to each finger is moved according to the finger angle stored in the array. When the motor moves, the lead screw fixed to each motor rotates, and the shaft fixing frame connected to the lead screw moves. Accordingly, the shaft of the wearable glove moves, the sensor value of the flex sensor is changed, and the sensor value is compared with the angle of each finger stored in the array. If the input finger angle and sensor value are not the same, the DC motor is put into motion until they are equal. At the same time, the movement direction of the arm is displayed by driving the vibration motor in the corresponding direction according to the movement direction of the arm stored in the array. Thereafter, the wearable gloves are moved until the last value of the stored array is input. When the sign language expression is finished, the wearable glove returns to its initial state and anticipates the input of the next sign language video. If there is no additional input video, the algorithm ends by terminating the motion of the wearable glove.

Figure 12 shows the result of expressing two words (dog, police) using 3D modeling through the structure and motion algorithm of the wearable glove described above. Figure 12a shows the initial state of the wearable glove. Figure 12b shows the movement of the wearable glove when a word corresponding to 'dog' is input. Figure 12c shows the wearable glove when a word corresponding to 'police' is entered. The two words (dog, police) shown in the picture are expressed as a picture because they were expressed in repeated motions from two to three times. When the initial state changes, the finger angle according to each sign language word is expressed as shown in Figure 12a–c. The lead screw was rotated by the rotation of the motor attached to the finger; thus, the shaft attached to each finger moves. Therefore, it is possible to express the sign language motion for the word being expressed in this way.

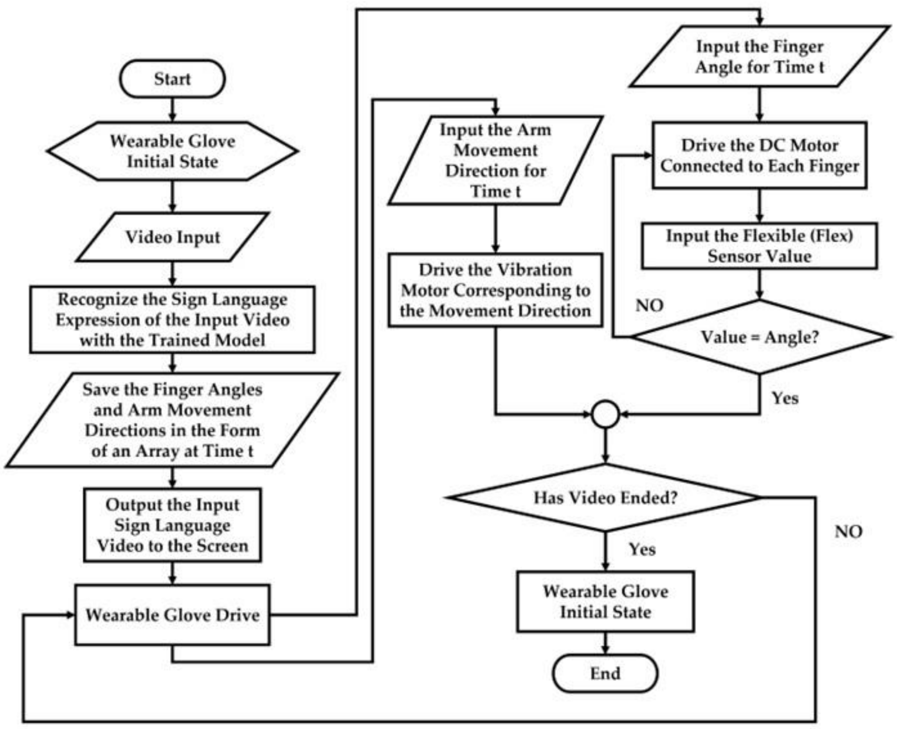

Start
Wearable Glove Initial State
Video Input
Recognize the Sign Language Expression of the Input Video with the Trained Model
Save the Finger Angles and Arm Movement Directions in the Form of an Array at Time t
Output the Input Sign Language Video to the Screen
Wearable Glove Drive
Input the Arm Movement Direction for Time t
Drive the Vibration Motor Corresponding to the Movement Direction
Input the Finger Angle for Time t
Drive the DC Motor Connected to Each Finger
Input the Flexible (Flex) Sensor Value
NO
Value = Angle?
Yes
Has Video Ended?
NO
Yes
Wearable Glove Initial State
End

Figure 11. Operation algorithm of the wearable glove.

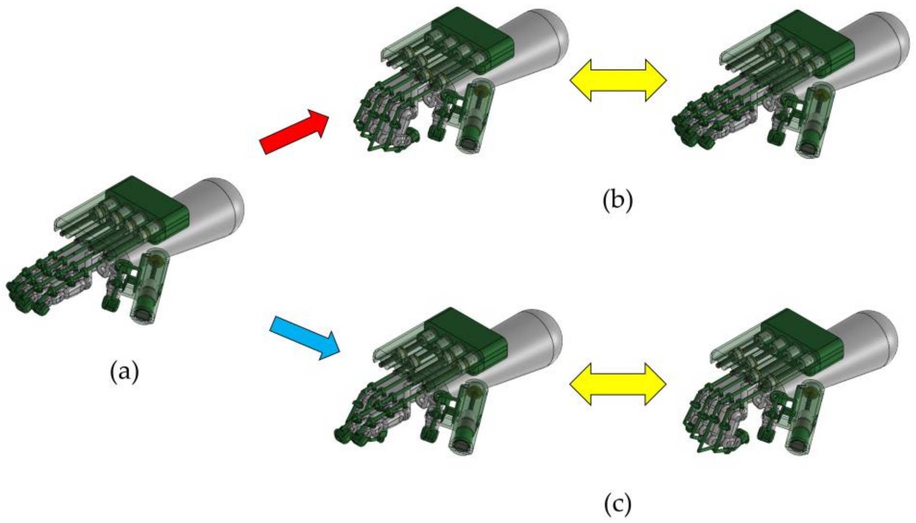


Figure 12. Operation of wearable glove using 3D modeling. (**a**) Initial state; (**b**) sign language word (dog) expression; (**c**) sign language word (police) expression.

3.2. Wearable Glove Prototyping

Figure 13 shows a prototype of the wearable glove, fabricated based on the design. Figure 13a shows the exoskeleton frame fixed to the glove, Figure 13b shows the fabricated prototype, and Figure 13c shows a person wearing the prototype and controlling the finger angle. As shown in Figure 13a, the exoskeleton frame was fixed to the upper surface of the glove, and interference between the frames was minimized. Through this, the wearer's discomfort due to interference between the frames was eliminated. As shown in Figure 13b, each shaft was fixed to the exoskeleton frame. The motor frame was also fixed to the glove and forearm. When the motor frame was fixed to the forearm, a length-adjustable vent was used to prevent the motor frame from moving. As shown in Figure 13c, it was confirmed that it is possible to control the finger angle with the shaft through the completed prototype.

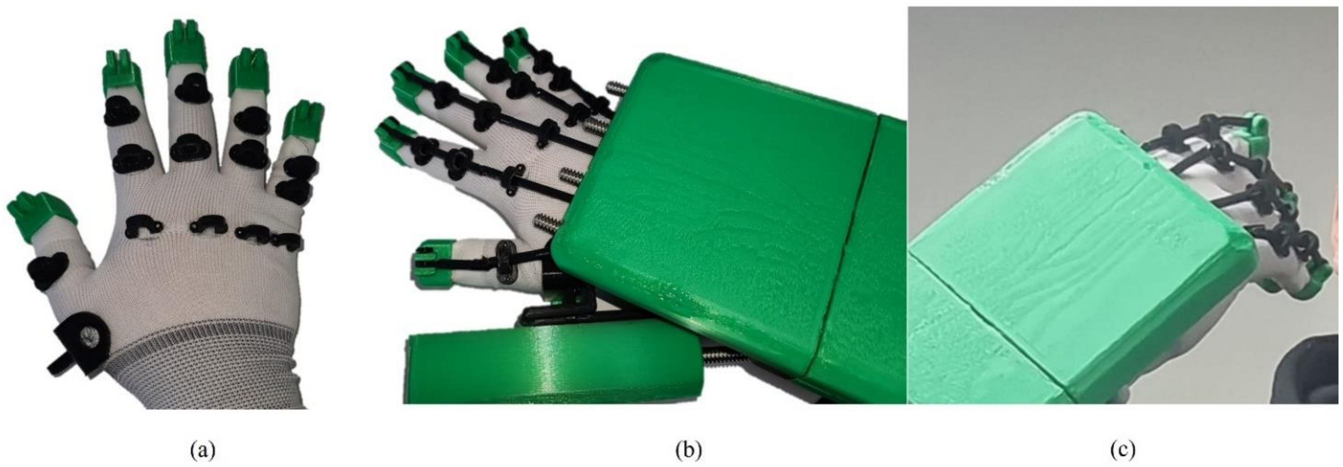


Figure 13. Prototype of the wearable glove: (**a**) glove with fixed exoskeleton frame; (**b**) fabricated prototype; (**c**) wearing the prototype and controlling the finger angle.

4. RNN and LSTM Performance Experiment

The activation function used for learning utilized the Rectified Linear Unit (ReLU) function and the Softmax function, the optimization function used the Adam function, and the loss function used the categorical cross-entropy function. The epoch was set to 300 so that training was repeated 300 times on all the data. The data used for training contained a total of 1200 videos. The validation data were trained by randomly selecting 20% of the total data. That is, 960 videos were used for training data, and 240 videos were used for the validation data. Additionally, 100 videos that were not used for training were selected as test data to evaluate the performance of the trained model.

The experimental environment is shown in Table 4. For data preprocessing, the same skeletal data were extracted using the MediaPipe Holistic API. The data used in the experiment differed only in the length of the sequence, and the other conditions were kept the same. We compared the training results of the RNN and LSTM using the generated dataset.

Table 4. Experimental environment.

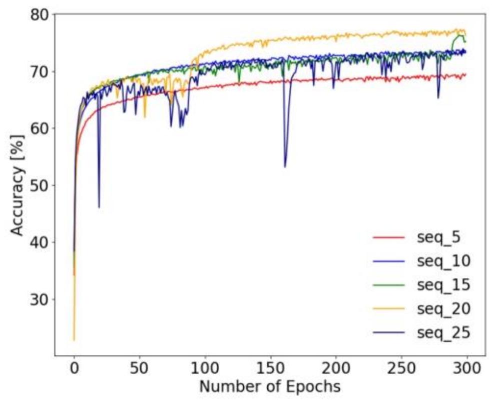
Sequence Length of Data	Sign Language Words	Number of Data
5	20	60
10	20	60
15	20	60
20	20	60
25	20	60

Table 5 shows the results associated with the creation of a dataset for the length of each sequence in the experimental environment, which is described in Table 4. As shown in Table 5, the total number of data points without sequence information is equal to 178,353. When data are created by dividing the information of each sequence, 178,253 data points for sequence 5, 178,153 data points for sequence 10, 178,053 data points for sequence 15, 177,953 data points for sequence 20, and 177,853 data points for sequence 25 are created. As a result, data differences according to sequence differences emerge, and it was confirmed that the difference decreases by 100 pieces of data whenever the sequence increases by 5.

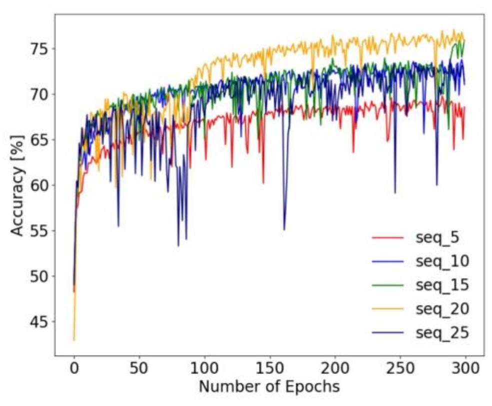
Table 5. Data sequence number and total number of preprocessed data points.

Sequence Length of Data	Total Number of Preprocessed Data Points
0	178,353
5	178,253
10	178,153
15	178,053
20	177,953
25	177,853

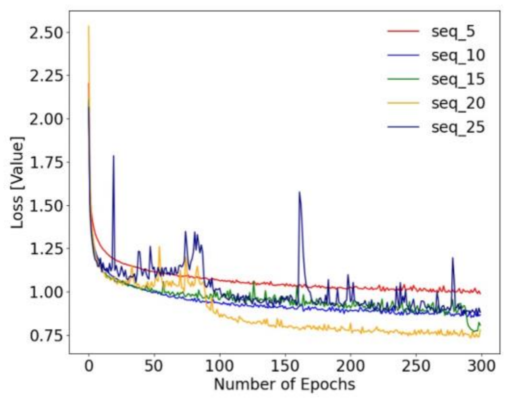
Figure 14 shows the results of training with the RNN based on the dataset generated according to the sequence. Figure 14 shows: (a) the accuracy of the training data, (b) the accuracy of the validation data, (c) the loss of the training data, and (d) the loss of the validation data. As shown in Figure 14, the smaller the training data sequence, the more stable the result. Looking at Figure 14a,b, it can be seen that as the length of the sequence increases, the accuracy rapidly drops during the training process and then rises again. Additionally, as the sequence lengthens, accuracy increases and loss decreases. Looking at Figure 14c,d, it can be seen that the loss rapidly increases during the training process as the length of the sequence increases, and then the loss decreases again. However, it was confirmed that the longest sequence length of 25 has a lower accuracy and higher loss than the sequence length of 20. The RNN training result for the sequence length of 20 confirmed that the validation data accuracy was 75.75% and the validation data loss was 0.7575. The RNN training result for the sequence length of 25 confirmed that the validation data accuracy was 71.43% and the validation data loss was 0.7143. Accordingly, the best training result in the RNN model was obtained when the sequence length of data was 20.



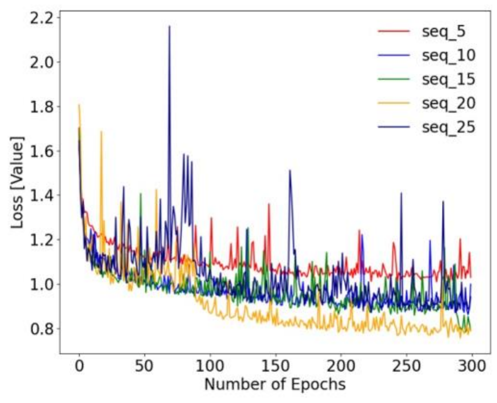
(a)



(b)



(c)



(d)

Figure 14. RNN training result according to data sequence: (**a**) training data accuracy, (**b**) validation data accuracy, (**c**) training data loss, and (**d**) validation data loss.

Figure 15 shows the results trained with the datasets created according to each sequence. Figure 15a–d shows the accuracy plots of the training data, the accuracy of the validation data, the loss of training data, and the loss of validation data, according to the sequences seq_5, seq_10, seq_15, seq_20, and seq_25, which correspond to the sequences of 5, 10, 15, 20, and 25, respectively. Figure 15a,b shows that the shorter the sequence length, the more stable the training accuracy; correspondingly, the longer the sequence length, the higher the training accuracy. The sequence length of 25 did not train the data, as confirmed in Figure 15c,d. The accuracy acc of the training data was 7.92%, the accuracy val_acc of the validation data was 7.85%, the loss of the training data was 2.98, and the loss val_loss of the validation data was also 2.98. It was determined that this lack of training was related to the 25-sequence length of the training data. As shown, the shorter the sequence length, the more stable the loss, and the longer the sequence length, the smaller the loss. As indicated, the sequence length of 25 has a higher loss than other sequences.

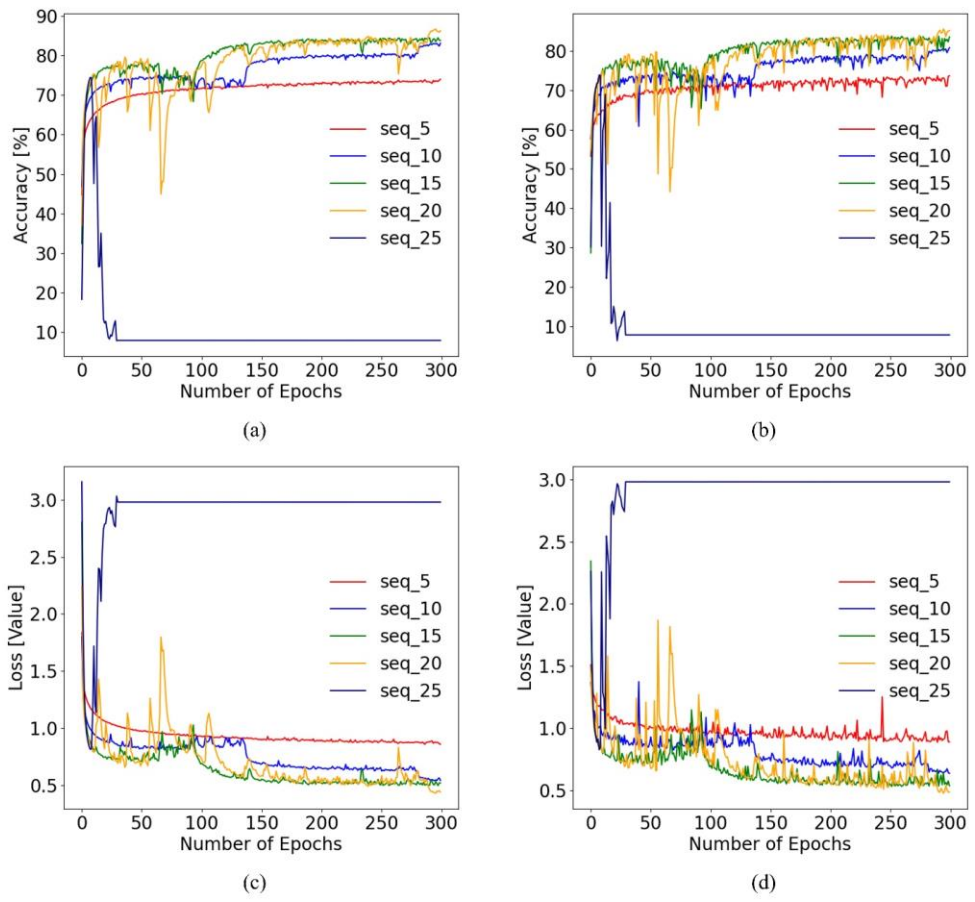


Figure 15. LSTM training result according to data sequence: (**a**) training data accuracy, (**b**) validation data accuracy, (**c**) training data loss, and (**d**) validation data loss.

Table 6 lists the values obtained after completing the training of the RNN and LSTM according to each sequence. The listings train_acc, val_acc, train_loss, and val_loss represent the training data accuracy, validation data accuracy, training data loss, and validation data loss, respectively. When the data sequence length is 20, the accuracy of RNN validation data is 75.75% and LSTM validation data is 85.14%, showing the best performance. Referring to Table 6 and Figures 14 and 15, the LSTM model using the dataset with a sequence length of 20, which output more accurate and relatively stable training results than the other sequences, was selected as the sign language recognition model for the wearable glove.

Table 6. Training results obtained for different data sequences.

Model	Sequence Length of Data	Train_acc	Val_acc	Train_loss	Val_loss
RNN	5	69.44%	68.57%	0.9894	1.0298
	10	73.51%	71.06%	0.8669	0.9991
	15	75.21%	75.93%	0.8072	0.7895
	20	76.28%	75.75%	0.7818	0.7575
	25	73.29%	71.43%	0.8812	0.7143
LSTM	5	73.95%	73.65%	0.8575	0.8891
	10	83.09%	80.74%	0.5410	0.6357
	15	83.81%	83.49%	0.5186	0.5354
	20	86.25%	85.14%	0.4386	0.4847
	25	7.92%	7.85%	2.9812	2.9813

Table 7 shows the results using the test data of an LSTM model trained using a data sequence length of 20. Table 7 shows the results of the test data of the selected models. The number of correct answers and the number of incorrect answers for each word are indicated. For the test data, five videos were selected for each word; thus, there was a total of 100 videos. It was confirmed that there was one incorrect answer for most words. As a result of analyzing the data with incorrect answers, it was confirmed that the error rate for videos shot from the side was large. As a result, the accuracy of the test data was confirmed to be 85%.

Table 7. Test results of an LSTM model trained on a data sequence length of 20.

Sign Language Word	Number of Videos	Correct Answers	Incorrect Answer
Dog	5	5	0
Police	5	5	0
Stairs	5	4	1
Moon	5	4	1
Bat	5	5	0
Bee	5	4	1
Hospital	5	4	1
Bandage	5	4	1
Teacher	5	4	1
Baby	5	4	1
Apartment	5	4	1
Dizziness	5	5	0
Elevator	5	4	1
Glass	5	4	1
Food	5	4	1
Car	5	5	0
Toy	5	4	1
Thermometer	5	4	1
Friend	5	4	1
Toilet	5	4	1
Total	100	85	15

5. Discussion

This paper proposes a wearable glove to assist in learning Korean sign language. A wearable glove using a DC motor was designed to assist sign language learning, and sign language training was conducted using ANN models to express sign language in the wearable glove. As ANN models for training, RNN and LSTM models were selected. As a result of Korean sign language training, the RNN model and the LSTM model confirmed 85.14% recognition rate accuracy for the validation data.

In this section, a comparative analysis of related studies on recognizing and expressing sign language using wearable hands was additionally prepared. We compared the performance results of systems related to sign language recognition using sensor signals from wearable gloves or sign language video data. Table 8 shows the comparison of sign language recognition systems.

Table 8. Comparison of sign language recognition systems.

Author	Method	Language Level	Architecture	Accuracy
Wu, J [7]	Sensor (IMU, sEMG sensor)	Words	SVM	96.16%
Lee, B. G [8]	Sensor (IMU, Flex sensor, Pressure sensor)	Alphabets	SVM	98.2%
Abhishek, K. S [9]	Sensor (Touch sensor)	Alphabets, Numbers	None (Classification based on touch sensor signals)	92%
Zhou, Z [10]	Sensor (YSSA)	Hand Gestures	SVM	98.63%
Choi, S. G [19]	Video	Gestures	ResNet	84.5%
This work	Video	Gestures	LSTM	85.14%

First, there are methods for recognizing ASL based on sensor data, and these studies commonly show high accuracy for sign language recognition [7–10].

There is a study on recognizing sign language words using data from IMU and sEMG sensors [7]. In [7], 80 ASL words were recognized through machine learning based on sensor data. In the machine learning process, decision tree, support vector machine (SVM), nearest neighbor, and Naïve Bayes classifiers were used. Among them, when the SVM with the highest accuracy was selected, an accuracy of 96.16% was obtained.

There is a study on recognizing 28 gestures including 26 letters of the alphabet, neutral state and an invalid sign using IMU, flex sensor, and pressure sensor data [8]. In [8], machine learning was performed using the data acquired through the sensor of the wearable glove, and the machine learning model used SVM. As a result, the machine learning results through SVM showed an accuracy of 98.16%.

There is a study on recognizing 26 letters of the alphabet and numbers from 0 to 9 using data from a touch sensor attached to a wearable glove [9]. In [9], alphabets and numbers were recognized by classifying the touch sensor data collected without AI learning, and an accuracy of 92% was obtained.

There is a study on recognizing 11 hand gestures using YSSA data [10]. In [10], sign language information was obtained through YSSA, which has high elasticity and a fast response speed, and it was recognized using SVM. Additionally, taking advantage of the fast response speed of sign language recognition, the recognition accuracy of hand gestures was 98.63% in a short time of less than 1 s.

On the other hand, the sign language recognition learning method using video data showed relatively low accuracy compared to studies related to sensor-based sign language recognition. Therefore, in this study, the performance was compared with studies directly related to expressing Korean sign language. There is a study on recognizing Korean sign language gestures based on video data of Korean sign language [19]. In [19], the size of the dataset was reduced to solve the disadvantage of the machine learning method based on video data, which is that the data size is large and difficult to manage. In addition, as a result of training using the residual neural network (ResNet) using the reduced data, an accuracy of 84.5% was obtained.

This study shows low accuracy compared to studies that recognize sign language using sensor data. However, when compared to previous studies that learned Korean sign language with video data, it shows relatively higher accuracy. It seems that, if more sign language words are learned, the correct answer rate can be additionally increased. The accuracy and correct answer rate are expected to be further improved by increasing the amount of sign language words and using the improved GPU. In addition, if these limitations are resolved, Korean sign language can be recognized with higher accuracy, which is expected to be of great help in developing devices for assisting sign language learning.

6. Conclusions

This paper proposes a wearable glove for Korean sign language education using an LSTM model. The external appearance of a wearable glove was designed for sign language education, as well as its operation algorithm. A prototype was produced based on the designed wearable glove and operating algorithm, and it was confirmed that it was possible to control the finger angle. To train the deep learning model, we designed a training dataset with 20 Korean sign language words. Using this, RNN and LSTM training was conducted. In addition, when using a training data sequence length set to 25, the LSTM model was not trained. To solve this problem, an experiment was conducted to compare the difference in the total number of data according to the sequence length of the training data and the training accuracy and loss of the RNN and LSTM and determine this difference. It was confirmed that the smaller sequences are stable but relatively less accurate, and larger sequences are unstable but have higher accuracy. Based on the results of the RNN and LSTM training experiments, a model that can be used for sign language recognition using wearable gloves was selected. The selected model corresponds to a data sequence length of 20 for the LSTM, and the correct answer rate was 85%.

In the future, we aim to improve the expression and accuracy of more words by training with more than 100 sign languages. When a sentence is input based on the trained sign language words, it will be possible to continuously convert the input sentence into a sign language sentence and express this sentence. In addition, we plan to improve the structural design of the wearable glove and the control method that improves response performance so that it can achieve a better sign language learning performance than the current model.

Author Contributions: Conceptualization, H.-J.K.; methodology, H.-J.K.; validation, S.-W.B.; investigation, H.-J.K.; writing—original draft preparation, H.-J.K.; writing—review and editing, S.-W.B.; supervision, S.-W.B.; project administration, S.-W.B.; funding acquisition, S.-W.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by a 2021 Research Grant from Sangmyung University (2021-A000-0263).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lee, H.H.; Song, M.Y.; Hong, S.E.; Lee, E.Y.; Kang, C.W.; Won, S.O. A Study on the Characteristics of Sequential Combination Structure of Korean Signs-Focused on the Seoul Data of the Korean Sign Language Corpus. *Korean Soc. Educ. Hear. Lang. Impair.* **2020**, *11*, 117–199. [CrossRef]
2. Shin, H.I. A Bridge between Meaning and Form: Implications of Iconicity for Korean Sign Language Learning. *Asian J. Educ.* **2019**, *20*, 301–320. [CrossRef]
3. Kim, H.J. A Deaf People's Perspective on Deaf Identity. *Korean Soc. Educ. Hear. Lang. Impair.* **2021**, *12*, 47–65. [CrossRef]
4. Choi, S.K. Deaf People's Own Perspective with Participants to Education for Students with Hearing Impairment. *Korean J. Political Sci.* **2020**, *28*, 145–170. [CrossRef]
5. Caselli, N.K.; Hall, W.C.; Henner, J. American Sign Language Interpreters in Public Schools: An Illusion of Inclusion that Perpetuates Language Deprivation. *Matern. Child Health J.* **2020**, *24*, 1323–1329. [CrossRef] [PubMed]
6. Cheok, M.J.; Omar, Z.; Jaward, M.H. A review of hand gesture and sign language recognition techniques. *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 131–153. [CrossRef]
7. Wu, J.; Sun, L.; Jafari, R. A Wearable System for Recognizing American Sign Language in Real-Time Using IMU and Surface EMG Sensors. *IEEE J. Biomed. Health Inform.* **2016**, *20*, 1281–1290. [CrossRef]

8. Lee, B.G.; Lee, S.M. Smart Wearable Hand Device for Sign Language Interpretation System with Sensors Fusion. *IEEE Sens. J.* **2018**, *18*, 1224–1232. [CrossRef]
9. Abhishek, K.S.; Qubeley, L.C.F.; Ho, D. Glove-based hand gesture recognition sign language translator using capacitive touch sensor. In Proceedings of the 2016 IEEE International Conference on Electron Devices and Solid-State Circuits (EDSSC), Hong Kong, China, 3–5 August 2016; pp. 334–337. [CrossRef]
10. Zhou, Z.; Chen, K.; Li, X.; Zhang, S.; Wu, Y.; Zhou, Y.; Meng, K.; Sun, C.; He, Q.; Fan, W.; et al. Sign-to-speech translation using machine-learning-assisted stretchable sensor arrays. *Nat. Electron.* **2020**, *3*, 571–578. [CrossRef]
11. Kim, K.W.; Lee, M.S.; Soon, B.R.; Ryu, M.H.; Kim, J.N. Recognition of sign language with an inertial sensor-based data glove. *Technol. Health Care* **2016**, *24*, S223–S230. [CrossRef]
12. Wadhawan, A.; Kumar, P. Deep learning-based sign language recognition system for static signs Sensors Fusion. *Neural Comput. Appl.* **2020**, *32*, 7957–7968. [CrossRef]
13. Al-Hammadi, M.; Muhammad, G.; Abdul, W.; Alsulaiman, M.; Bencherif, M.A.; Mekhtiche, M.A. Hand Gesture Recognition for Sign Language Using 3DCNN. *IEEE Access* **2020**, *8*, 79491–79509. [CrossRef]
14. Mariappan, H.M.; Gomathi, V. Indian Sign Language Recognition through Hybrid ConvNet-LSTM Networks. *EMITTER Int. J. Eng. Technol.* **2021**, *9*, 182–203. [CrossRef]
15. Samaan, G.H.; Wadie, A.R.; Attia, A.K.; Asaad, A.M.; Kamel, A.E.; Slim, S.O.; Abdallah, M.S.; Cho, Y.-I. MediaPipe's Landmarks with RNN for Dynamic Sign Language Recognition. *Electronics* **2022**, *11*, 3228. [CrossRef]
16. Ismail, M.H.; Dawwd, S.A.; Ali, F.H. Dynamic hand gesture recognition of Arabic sign language by using deep convolutional neural networks. *Indones. J. Electr. Eng. Comput. Sci.* **2020**, *25*, 952–962. [CrossRef]
17. Kothadiya, D.; Bhatt, C.; Sapariya, K.; Patel, K.; Gil-González, A.-B.; Corchado, J.M. Deepsign: Sign Language Detection and Recognition Using Deep Learning. *Electronics* **2022**, *11*, 1780. [CrossRef]
18. Abdullahi, S.B.; Chamnongthai, K. American Sign Language Words Recognition of Skeletal Videos Using Processed Video Driven Multi-Stacked Deep LSTM. *Sensors* **2022**, *22*, 1406. [CrossRef]
19. Choi, S.-G.; Park, Y.; Sohn, C.-B. Dataset Transformation System for Sign Language Recognition Based on Image Classification Network. *Appl. Sci.* **2022**, *12*, 10075. [CrossRef]
20. Nihal, R.A.; Broti, N.M.; Deowan, S.A.; Rahman, S. Design and Development of a Humanoid Robot for Sign Language Interpretation. *SN Comput. Sci.* **2021**, *2*, 220. [CrossRef]
21. Meghdari, A.; Alemi, M.; Zakipour, M.; Kashanian, S.A. Design and Realization of a Sign Language Educational Humanoid Robot. *J. Intell. Robot. Syst.* **2019**, *95*, 3–17. [CrossRef]
22. Al-khazraji, S.; Berke, L.; Kafle, S.; Yeung, P.; Huenfauth, M. Modeling the Speed and Timing of American Sign Language to Generate Realistic Animations. In Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility, Galway, Ireland, 22–24 October 2018; pp. 259–270. [CrossRef]
23. Sreelakshmi, M.; Subash, T.D. Haptic Technology: A comprehensive review on its applications and future prospects. *Mater. Today Proc.* **2017**, *4*, 4182–4287. [CrossRef]
24. Liu, L.M.; Li, W.; Dai, J.J. Haptic technology and its application in education and learning. In Proceedings of the2017 10th International Conference on Ubi-Media Computing and Workshops (Ubi-Media), Pattaya, Thailand, 1–4 August 2017; pp. 1–6. [CrossRef]
25. Medellin Castillo, H.I.; Zaragoza Siqueiros, J.; Govea Valladares, E.H.; Garza Camargo, H.; Lim, T.; Ritchie, J.M. Haptic-enabled virtual training in orthognathic surgery. *Virtual Real.* **2021**, *24*, 53–67. [CrossRef]
26. Lee, S.H. Research and development of haptic simulator for Dental education using Virtual reality and User motion. *Int. J. Adv. Cult. Technol.* **2018**, *6*, 52–57. [CrossRef]
27. Pala, F.K.; Türker, P.M. Developing a haptic glove for basic piano education. *World J. Educ. Technol. Curr. Issues* **2019**, *11*, 38–47. [CrossRef]
28. Marchal Crespo, L.; Raai, M.V.; Rauter, G.; Wolf, P.; Riener, R. The effect of haptic guidance and visual feedback on learning a complex tennis task. *Exp. Brain Res.* **2013**, *231*, 277–297. [CrossRef]
29. Gao, L.; Li, H.; Liu, Z.; Liu, Z.; Wan, L.; Feng, W. RNN-Transducer based Chinese Sign Language Recognition. *Neurocomputing* **2021**, *434*, 45–54. [CrossRef]
30. MediaPipe Holistic. Available online: https://google.github.io/mediapipe/solutions/holistic.html (accessed on 21 November 2022).
31. MediaPipe Hands. Available online: https://google.github.io/mediapipe/solutions/hands.html (accessed on 21 November 2022).
32. MediaPipe Pose. Available online: https://google.github.io/mediapipe/solutions/pose.html (accessed on 21 November 2022).
33. AI-Hub. Available online: https://www.aihub.or.kr/aihubdata/data/view.do?currMenu=120&topMenu=100&aihubDataSe=extrldata&dataSetSn=264 (accessed on 21 November 2022).