Wireless Communications and Mobile Computing

Gesture-Based Interfaces for Recognition and Control

Lead Guest Editor: Hugo Landaluce
Guest Editors: Laura Arjona and Vaishnavi Ranganathan

Gesture-Based Interfaces for Recognition and Control

Wireless Communications and Mobile Computing

Gesture-Based Interfaces for Recognition and Control

Lead Guest Editor: Hugo Landaluce
Guest Editors: Laura Arjona and Vaishnavi Ranganathan

Chief Editor

Associate Editors

Academic Editors

Contents

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 6879206, 11 pages
https://doi.org/10.1155/2022/6879206

Research Article

Motion Gesture Delimiters for Smartwatch Interaction

Yiming Zhao[1], **Yanchao Zhao**[1], **Huawei Tu**[2], **Qihan Huang**[1], **Wenlai Zhao**[1] and **Wenhao Jiang**[1]

[1]*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, 211100 Jiangsu, China*
[2]*Department of Computer Science and Information Technology, La Trobe University, Melbourne, Victoria 3086, Australia*

Correspondence should be addressed to Huawei Tu; h.tu@latrobe.edu.au

Received 25 January 2022; Revised 15 April 2022; Accepted 22 June 2022; Published 12 July 2022

Academic Editor: Laura Arjona

Smartwatches are increasingly popular in our daily lives. Motion gestures are a common way of interacting with smartwatches, e.g., users can make a movement in the air with their arm wearing the watch to trigger a specific command of the smartwatch. Motion gesture interaction can compensate for the small screen size of the smartwatch to some extent and enrich smartwatch-based interactions. An important aspect of motion gesture interaction lies in how to determine the start and end of a motion gesture. This paper is aimed at selecting gestures as suitable delimiters for motion gesture interaction with the smartwatch. We designed six gestures ("shaking wrist left and right," "shaking wrist up and down," "holding fist and opening," "turning wrist clockwise," "turning wrist anticlockwise," and "shaking wrist up") and conducted two experiments to compare the performance of these six gestures. Firstly, we used dynamic time warping (DTW) and feature extraction with KNN (*K*-nearest neighbors) to recognize these six gestures. The average recognition rate of the latter algorithm for the six gestures was higher than that of the former. And with the latter algorithm, the recognition rate for the first three of the six gestures was greater than 98%. According to experiment one, gesture 1 (shaking wrist left and right), gesture 2 (shaking wrist up and down), and gesture 3 (holding fist and opening) were selected as the candidate delimiters. In addition, we conducted a questionnaire data analysis and obtained the same conclusion. Then, we conducted the second experiment to investigate the performance of these three candidate gestures in daily scenes to obtain their misoperation rates. The misoperation rates of two candidate gestures ("shaking wrist left and right" and "shaking wrist up and down") were approximately 0, which were significantly lower than that of the third candidate gesture. Based on the above experimental results, gestures "shaking wrist left and right" and "shaking wrist up and down" are suitable as motion gesture delimiters for smartwatch interaction.

1. Introduction

Smartwatches have become a popular device in people's daily life [1]. People can use smartwatches in many day-to-day activities such as checking emails and sending and receiving messages [2]. Besides, smartwatches are also convenient for health management, e.g., sleep and heart rate monitoring [3, 4].

The questions of how to improve smartwatch interaction has attracted much attention in the HCI field. Currently, most popular commercial smartwatches such as Apple Watch still rely on touch interaction, physical buttons, and voice input [5]. These interaction methods are limited by screen size and environments, restricting the application of smartwatches to a wider extent. Therefore, smartwatches need new interaction methods to improve usability [6].

Motion gestures have potential advantages for smartwatch interaction [7]. For example, a user can draw a circle in the air with the wrist wearing a smartwatch to trigger a specific command of the smartwatch. Compared with interaction methods such as touchscreens, motion gesture interaction is less likely to be limited by the size of the screen. However, motion gesture interaction needs to address two main challenges. The first one is how to effectively obtain motion gesture data. Popular motion gesture recognition systems rely on cameras to capture gesture images or sensors such as gyroscopes and accelerometers to collect user action data [8–11]. Since smartwatches are mainly worn on the

wrist and move along with the wrist, we can use in-built sensors of smartwatches to collect gesturing data. Compared to gesture images captured using a camera, sensor data requires fewer computational resources to collect and can be used to identify gesture delimiters more effectively. The second one is how to determine the start and end of an intended gesture [12]. In the process of motion gesture interaction, the smartwatch needs to continuously record movement data, both nonuser-intended (e.g., the wrist keeps swinging while walking) and user-intended (performing defined gestures). Therefore, we need to specify the start and end of the intended gesture. There are two common ways. First, the user clicks the button to determine the start and end of a motion gesture [13], which usually requires the nonwatch-wearing hand to perform the click. This could interrupt the interaction flow. Second, the user performs a defined gesture as a delimiter. The defined delimiter is used to distinguish the gestures that the user intends to input from unintended ones. The delimiter should be significantly different from the common actions and other gestures to avoid false recognition and should be simple enough to perform. We use delimiters to determine the start and end of a gesture, which allows for a more natural way of user interaction and requires no additional hardware than using buttons.

This study is aimed at selecting suitable motion gestures as delimiters for smartwatches to improve smartwatch interaction in low power consumption and natural way. We first selected six candidate gestures: "shaking wrist left and right," "shaking wrist up and down," "holding fist and opening," "turning wrist clockwise," "turning wrist anticlockwise," and "shaking wrist up" (Figure 1). Then, we conducted two experiments to evaluate the performance of these gestures as motion gesture delimiters. Considering the relatively low computing power of the watch and the requirement for fast and stable delimiter recognition, we used DTW (dynamic time warping) [14] and feature extraction with KNN (K-nearest neighbors) [15] to perform gesture recognition based on the data collected by the inbuilt gyroscopes and accelerometers of smartwatches. Results showed that "shaking wrist left and right," "shaking wrist up and down," and "holding fist and opening" achieved significantly higher recognition rates than "turning wrist clockwise," "turning wrist anticlockwise," and "shaking wrist up." In addition, we conducted a usability evaluation to support this conclusion. Hence, we further evaluated the performance of the former three gestures in daily scenes in terms of misoperation rates. "Shaking wrist left and right" and "shaking wrist up and down" had a misoperation rate of approximately 0, which could be primarily considered as the delimiters for smartwatch interaction.

2. Related Work

2.1. Motion Gesture Data Collection and Gesture Recognition. Motion gesture data collection for wearable devices usually relies on sensors. For example, EMG sensors [16, 17] or pressure sensors [18] can be used to collect data generated by hand movements. However, current smartwatches do not have such sensors. Instead, it is common to use inbuilt sensors such as

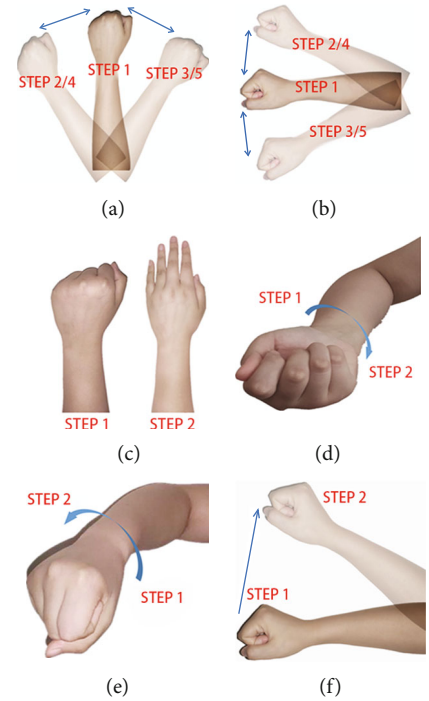


Figure 1: Six delimiter candidates. Gesture (a): "shaking wrist left and right"; gesture (b): "shaking wrist up and down"; gesture (c): "holding fist and opening"; gesture (d): "turning wrist clockwise"; gesture (e): "turning wrist anticlockwise"; and gesture (f): "shaking wrist up.".

accelerometers and gyroscopes to sense wrist movement and collect gesture data [19–21]. Our study also used these sensors for gesture data collection.

There are many methods for motion gesture recognition. Usually, feature extraction is carried out for gesture data, and neural network algorithms such as CNN, RNN, FNN, and HMM are trained for gesture recognition [22–26]. In our study, DTW (dynamic time warping) [14] and feature extraction + KNN(K-nearest neighbors) [15] are used for gesture recognition, respectively, as they need low sensor requirements and low computational requirements.

DTW can match two sequences of different lengths so that the minimum distance between the two sequences can be calculated. Then, the matching result can be compared based on this distance. The DTW algorithm has the advantages of short computation time. Sensors on smartwatches commonly collect gesture data at a fixed time interval. Two gesture data to be compared may have different lengths and cannot be matched directly. The DTW algorithm can be used to match data sequences of different lengths. And the distance and similarity between the two sequences of sample and template are calculated.

The feature extraction + KNN algorithm extracts some features of the whole set of gesture data and classifies the gesture data according to these features for gesture recognition. KNN calculates the distance between the sample X and the template samples and takes the top K template samples closest to it. If the top K samples have the most samples belonging to category R, then sample X also belongs to category R. K is usually an odd number not greater than 20.

Different K values generally lead to different classification results. Therefore, an optimal K value should be selected according to the results. The classification of template samples needs to be accurate as possible to ensure the correct classification of test samples.

2.2. Motion Gesture Interaction and Gesture Delimiters. Motion gestures are a promising way to interact with wearable devices [19, 27, 28]. Gesture interaction is especially suitable for mobile devices, such as changing the screen display direction by tilting the phone [29] and moving the cursor with gestures [30]. In addition, gesture interaction can also achieve more complex operations, such as text input with gestures on smartwatches [31], identity authentication by recognizing user gestures [32], and access data on virtual bookshelves around users [33].

An important step of using motion gestures is to separate normal smartwatch motion from a user's intended input. A common way to achieve it is to press button [13], but such a way requires both hands for interaction, which may not be always feasible. [34] collected IMU data to recognize three distinct phases of gesture entry: the start, middle, and end of a gesture motion for mobile devices. [35] used a dedicated delimiter sensor to detect the start and end of a gesture, which requires additional device support. [36] proposed a method for evaluating smartwatch delimiters using DTW, but using only accelerometers as data. Previous research has proposed to use double flip (i.e., rotating a smartphone along its long side to flip it twice) [12]. However, such a delimiter may not be applicable to wrist-worn devices, as it is in constant motion and therefore more error-prone. To find proper gesture delimiters for smartwatch interaction, our study considered six gesture candidates and examined their performance with two experiments.

3. Candidate Gesture Delimiters

Gesture delimiters applicable to smartwatch interaction should satisfy the following requirements:

(i) Easy to recognize: the smartwatch system needs to recognize gesture delimiters with high accuracy

(ii) Easy to learn: the user can learn gesture delimiters easily and recall them without much effort

(iii) Easy to perform: gesture delimiters would be performed frequently; so, they should have simple and should not lead to high hand and arm fatigue

To satisfy these requirements, six gestures were selected as candidate delimiters in our study. A pilot experiment with 6 right-handed participants was conducted to measure the average time of performing these gestures.

(i) Gesture 1: shaking wrist left and right. As shown in Figure 1(a), the user shakes the wrist twice with a small movement from side to side, and that the mean time of performing this gesture was 0.76 s

(ii) Gesture 2: shaking wrist up and down. As shown in Figure 1(b), the user slightly shakes his wrist twice from top to bottom. The average execution time of this gesture was 0.72 s

(iii) Gesture 3: holding fist and opening. As shown in Figure 1(c), the user clenches all fingers together and then opens them. The average execution time of this gesture was 0.55 s

(iv) Gesture 4: turning wrist clockwise. As shown in Figure 1(d), the user makes a fist and rotates the fist 90 degree clockwise. The average execution time was 1.13 s

(v) Gesture 5: turning wrist anticlockwise. As shown in Figure 1(e), the user makes a fist and rotates the fist 90 degree counterclockwise. The average time taken for gesture 5 was 1.10 s

(vi) Gesture 6: shaking wrist up. As shown in Figure 1(f), the user shakes the wrist upward significantly. Compared with the up-and-down shake of gesture 2, the movement range of gesture 6 is larger. The average time for performing gesture 6 was 0.53 s

Gestures 4 and 5 were designed based on the double flip gesture [12], which has been verified as a usable delimiter for mobile phone interaction. We would like to exam if gestures 4 and 5 would be feasible as a delimiter for smartwatch interaction.

4. Experiment One: Delimiter Recognition with DTW and Feature Extraction + KNN

We conducted an experiment to evaluate the effectiveness of the six delimiter candidates. The experiment consisted of two parts. First, we examined recognition rates of the six delimiters with DTW and feature extraction + KNN algorithms. Then, we evaluated the usability of the six delimiters according to subjective questionnaires.

4.1. Experimental Apparatus and Participants. The experiment was conducted with a Huawei Watch 2 smartwatch, which had a 1.2-inch round AMOLED display with a resolution of 390×390 pixels, a speed sensor, and a gyroscope sensor. The program was written in Python. In the experiment, sensor data of the smartwatch were recorded as gesture data, including the three axes of the acceleration and gyroscope sensors. Samples were recorded at 20 ms intervals.

There were a total of 10 participants (8 males) with an average age of 22.6 years. Each participant was asked to perform each gesture 30 times while wearing a smartwatch. Six of the students participated in the pilot study of gesture selection. Others did not have experience using smartwatches. All participants were right-handed and wore the smartwatch on their right hand. To obtain better results, we let the experimenter wear the watch with their dominant hand. Since the two hands are symmetrical, it should be reasonable to generalize the results from the right hand to the left hand.

In total, we collected 1800 gesture samples from 10 participants. We selected one sample per person per gesture for training, which means 60 samples for training and the remaining 1740 gestures for testing.

4.2. Data Preprocessing

4.2.1. Smoothing. Due to hand jitter and false operation of the user, the collected sensor data had a lot of noise. Figure 2(a) shows the raw data of gesture 1 collected from the x-axis of the accelerometer, which have many burrs and spikes. The burrs and spikes would reduce the accuracy and increase the difficulty of gesture segmentation and feature extraction. Smoothing filtering algorithms can reduce the noise.

There are many algorithms for smoothing filtering, e.g., moving average filter, median filter, and Gaussian filter. The algorithm of moving average filter was chosen in this study because it is relatively simple but effective. The moving average filter calculates the average value within a window and collects new data for each movement. The window slides forward, and the average value is calculated as the valid data. Figure 2(b) shows the data from Figure 2(a) after smoothing and filtering. It can be seen that after the moving average filter, the burrs and spikes in the data are effectively reduced.

4.2.2. Gesture Segmentation. In this experiment, we collected continuous three-axis acceleration sensors and three-axis gyroscope sensor data. The data collected by the sensors include unintended-gesture data and intended-gesture data. Gesture segmentation needs to extract valid gesture data from these data. Figure 3 shows part of the collected data of one trial of performing gesture 1. The data from 0 to 3, 4 to 6, and 7.5 to 8.5 s are not related to the delimiter, and the rest data is valid delimiter data. It can be seen that when the participant is performing the delimiter, the collected data is fluctuating significantly; when the participant is not performing the gesture, the collected data is fluctuating gently. So, we can rely on this feature to segment gestures.

We used differential methods to implement gesture segmentation. Differential methods can effectively show the volatility of data and have the advantages of easy implementation and running in real-time. A differential method is performed to obtain the total change data of two sensors, and then the start and end of the gesture are calculated by comparing the total change of two sensors with the preset threshold value. The main steps of gesture segmentation using the differential method are as follows.

Calculate the data variation: the formula for calculating the variation is as follows:

$$\Delta \mathbf{a}_k = |x_k - x_{k-1}| + |y_k - \mathrm{y}_{k-1}| + |z_k - \mathrm{z}_{k-1}|, \tag{1}$$

where x_k, y_k, and z_k represent the values of the sensor in the x, y, and z axis at the $k-$th data point, respectively. Since the data collected in this experiment come from two different sensors, it is necessary to calculate the variation of two sensors and add the two variations to obtain the total variation ΔA_k. To make the system more robust, we use the simple moving average algorithm (SMA) with a window size of 3 ($W = 3$) to smooth the gesture data. The mean of the k-th and subsequent $W - 1$ data points is denoted as SMA_k, which is the k-th data point after smoothing. SMA_k is calculated as Eq. (2).

$$\mathrm{SMA}_k = \frac{1}{W} \sum_{k}^{k+W} \Delta A_k. \tag{2}$$

Calculate the threshold: Figure 4 shows the data after differential processing in a trial. The results show the fluctuation of the data. The more intense the fluctuation, the more likely the data from valid gestures. For data lasting more than eighty seconds, the differential value fluctuations appear twenty times, and each fluctuation corresponds to the data variation of the gesture's six-axis data in Figure 3. To effectively identify the valid gesture interval, two thresholds need to be set: "Start" represents the start threshold, and "End" represents the end threshold. And to filter out the noise and the integrity of the gesture data, "Start" should be greater than "End."

Since the fluctuation range of gestures is different, the thresholds of gestures are also different. The threshold values selected for gestures are calculated based on our experimental data, as shown in Table 1. For example, when we segment the differential data of gesture 1 in Figure 4, we first detect the differential value 0.9 as the beginning of the gesture, and then we mark the end of the gesture when the differential value drops to 0.6. Generally, the greater the fluctuation range of a gesture, the greater the data variation, the larger difference between the "Start" and "End" thresholds, and vice versa.

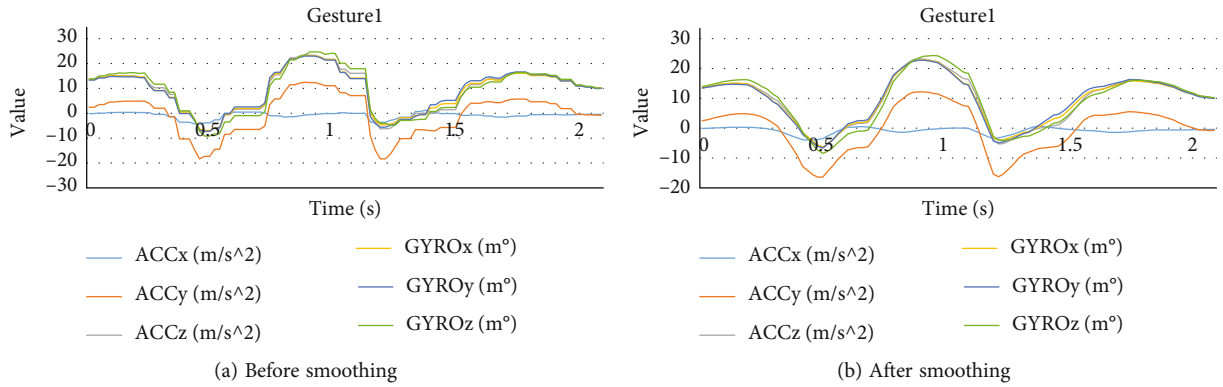
Result of segmentation is as follows: Figure 5(a) shows the fluctuation graph of a valid gesture after the segmentation of Figure 3. Figure 5(b) shows the fluctuation graph of a single valid gesture data for gesture 2. It can be seen that the trends of the sensor data for gesture 1 and 2 are different due to the different trajectories of gesture movement. Hence, we can perform gesture recognition based on the data characteristics.

In addition to the data trend, features such as mean, variance, and peak-to-peak values can also reflect the differences in this data. Figure 6 shows the mean and variance between gestures 1 and 2 on different axes. The data for gestures 1 and 2 are very different, except for the average values over the acceleration z-axis. Hence, we can recognize gestures based on such feature differences.

4.3. Classification Methods.

This study uses both DTW and feature extraction with KNN methods for gesture recognition. Although these are traditional methods, they are easy to implement and suitable for fast recognition of delimiters on smartwatches with low computational power. Future work will consider other algorithms to cater for other requirements of gesture interaction (e.g., higher recognition accuracy).

4.3.1. DTW. Dynamic time warping (DTW) is a simple recognition algorithm based on the idea of dynamic



(a) Before smoothing (b) After smoothing

Figure 2: Gestures before and after smoothing.

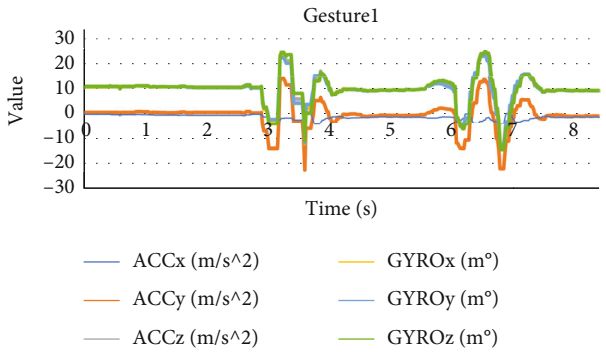


Figure 3: Part of the data collected in an experiment.

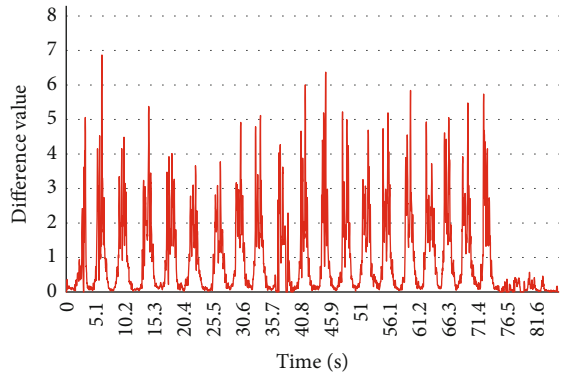


Figure 4: Differential processing result.

Table 1: Gesture threshold.

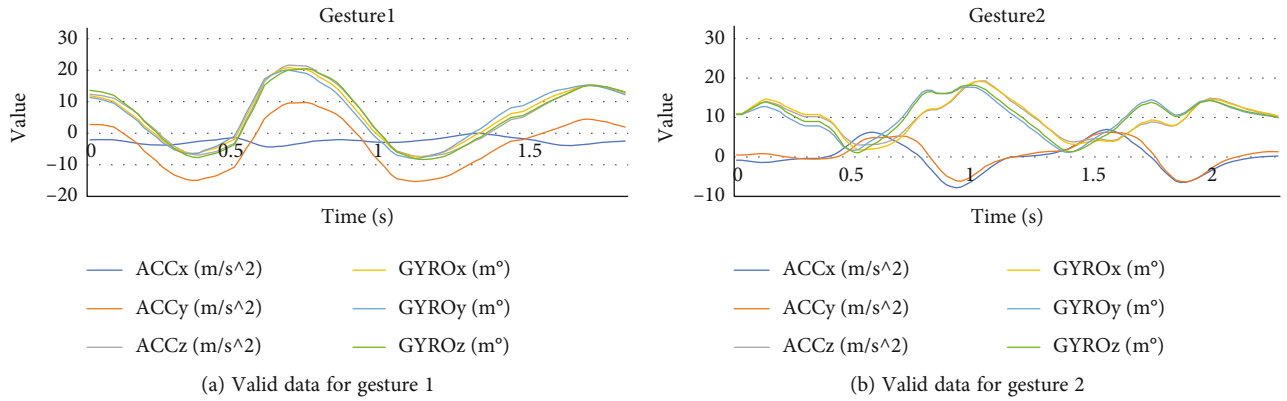
Gesture	Start	End
Gesture 1	0.9	0.6
Gesture 2	1.1	0.4
Gesture 3	0.56	0.1
Gesture 4	0.27	0.18
Gesture 5	0.2	0.1
Gesture 6	1	0.2

programming to solve the matching problem of two different length sequences. It calculates the minimum distance between two samples to match and recognize. DTW is widely used in speech recognition, which has the advantages of low computing time cost and few templates compared with other recognition algorithms. As with speech, the data generated by each person performing the gesture is different. Different performing time leads to different length data and not-strict linear correspondence. Direct matching of templates and samples of different lengths is not available; so, we use dynamic warping for the sequence to solve this problem.

First, the system creates an $N \times M$ matrix D, where the number of rows N represents the number of frames of the sample sequence to be recognized, and the number of columns M represents the number of frames of the template sequence, i.e., the sample sequence to be identified is $T = [T_1, T_2 \ldots T_{n-1}, T_n]$, and the template sequence is $R = [R_1, R_2 \ldots R_{m-1}, R_m]$. T_n is the feature of the n-th frame with the frame length f. Similarly, R_m represents the feature of the m-th frame of the template with the frame length f. Since this experiment collects six-axis data from two sensors, the length of each frame is 6, i.e., $f = 6$, which represents the acceleration three-axis coordinate and gyroscope three-axis coordinate. D_{ij} represents the shortest distance between node i of T and node j of R. The D_{NM} is the shortest distance between two sequences.

Then, we calculate D_{ij}. Since the two sample sequences are not equal in length, we use the nonlinear matching method in DTW. As shown in Figure 7, we take T as the horizontal axis and R as the vertical axis and draw a grid diagram in the coordinate system. The intersection points in the grid represent the distance between the template at frame m and the sample to be identified at frame n. We find the shortest path from $(0, 0)$ to (N, M) based on the thought of dynamic programming. The point (i_n, j_m) is the optimal point decided by (i_{n-1}, j_m), (i_n, j_{m-1}), (i_{n-1}, j_{m-1}), and the best choice based on the previous section of the path. Thus, we have $D(i, j)$ as follows:

$$D(i_n, j_m) = \min \{D(i_{n-1}, j_m), D(i_n, j_{m-1}), D(i_{n-1}, j_{m-1})\} + d(i_n, j_m), \quad (3)$$



(a) Valid data for gesture 1

(b) Valid data for gesture 2

Figure 5: Valid data for gestures 1 and 2.

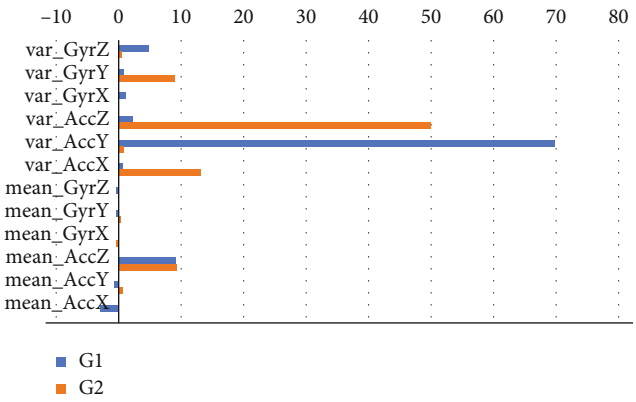


Figure 6: Variance and mean of gesture 1 and gesture 2.

where $d(i_n, j_m)$ is the distance between two sequences at (m, n). This experiment uses the 2-norm algorithm, the Euclidean distance, to calculate the distance between two vectors.

The short distance D_{NM} between the template R and the sample T indicates that the template R has a high similarity with T, which means they may come from the same gesture set. By calculating the shortest distance between the sample to be detected and multiple templates, we can get the best matching gesture based on the maximum similarity, i.e., the shortest matching distance. The algorithmic process of gesture recognition using DTW is shown in Figure 8.

4.3.2. K-Nearest Neighbor. The K-nearest neighbor algorithm (KNN) is a simple method in data mining, and its key idea is that if a sample has K nearest samples, most of which belong to class R, then the sample also belongs to class R. The selection of K has a significant impact on the overall classification result; so, an optimal K value should be selected based on comparative experiments, and K is usually an odd number no greater than 20.

In general, the label of the template data is known, but the label of the test sample is unknown. The system calculates distance (similarity) between the test samples and the templates by Euclidean distance and selects K nearest samples. Based on the class of K nearest samples, the system finds the most occurring class R, which is the label of the test sample. The algorithm described as follows:

(1) Calculating the distance between the test sample and the template data

(2) Sorting in ascending order by distance

(3) Selecting the K nearest samples

(4) Calculating the occurrence frequency of the class of K nearest samples

(5) According to the class and occurrence frequency of K nearest samples, the class R with the highest occurrence frequency is selected, which is the class of test sample

4.4. Results

4.4.1. Result Analysis of DTW. After the preprocessing step in Section 5.2, we used the DTW algorithm to perform gesture recognition. The DTW algorithm depends on the results of matching with templates. To eliminate recognition errors caused by inaccurate templates, we used 10 templates per gesture and took the average value. That is, there were 10 templates R for each gesture, and we need to calculate the distance of the test gesture T from these templates R and take the mean values of them as the final distance between the test gesture T and the training sample. The DTW algorithm is shown in Figure 7.

The DTW algorithm can easily recognize gestures, but it has a high computational cost. Although the time for DTW to recognize a single gesture data is short, it took significantly longer times when the amount of gesture data increases. In this experiment, 100 samples were collected for each gesture, and it took about 3 minutes to calculate the recognition result for each gesture data set. Although the approximate FastDTW algorithm can be considered, it reduces the running time at the expense of lowering recognition rate. In order to achieve a high recognition rate and short recognition time for a single gesture, the classic DTW algorithm was used in this study.

Figure 9 shows the recognition rate of each gesture. Since the motion range of gesture 3 is relatively slight, it is easily

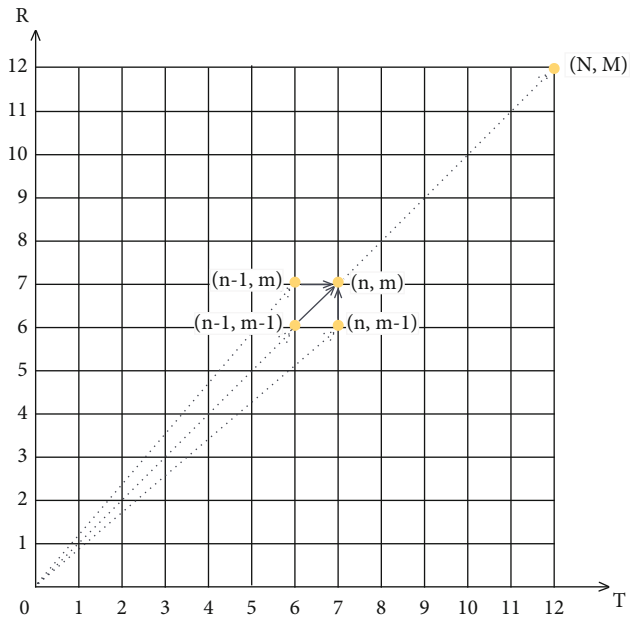


Figure 7: Nonlinear matching method in DTW.

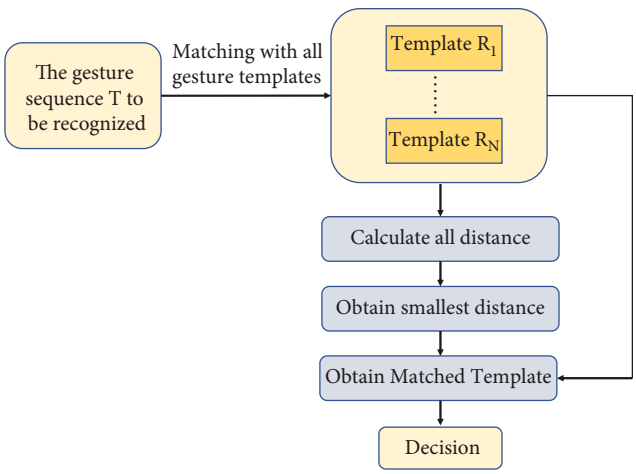


Figure 8: The process of DTW to recognize gestures.

confused with other gestures. After excluding gesture 3, Figure 10 shows the recognition rates of other gestures. It can be found that gesture 6 (shaking wrist up) and gesture 2 (shaking wrist up and down) are easily misidentified due to their high similarity in movement.

4.4.2. Result Analysis of Feature Extraction with KNN. After the preprocessing step in Section 5.2, we used the feature extraction + KNN algorithm for delimiter recognition. We needed to extract features from the data and then performed delimiter recognition based on the KNN algorithm.

We considered 30 features for our purpose. For x, y, and z axis of the accelerometer and gyroscope, we used the features of average value, variance, peak-to-peak, and interquartile. Besides, for the accelerometer and gyroscope, we used the correlation coefficient between x-axis and y-axis, between x-axis and z-axis, and between y-axis and z-axis. We then examined if using a subset of the 30 features could achieve similar recognition rates as using the 30 features. We employed the ExhaustiveSearch method provided on WEKA (Waikato Environment for Knowledge Analysis). This method finds a result with the highest recognition rate in the full set and all subsets. By combining it with evaluation strategies (CfsSubsetEval), we found that using all 30 features could obtain the highest recognition rate. Therefore, all the 30 features were adopted for further analysis with the KNN algorithm.

The K value of KNN has a significant impact on the experimental results, the estimation, and approximation errors. The K value is usually an small odd number to balance the estimation and approximation errors. The K value was set as 1, 3, 5, and 7 in this study. Table 2 shows the result of recognition rates for the six gestures with the four K values. After a comprehensive comparison, the gesture recognition rate with $K = 1$ should have the highest recognition accuracy. As shown in Figure 11, the recognition results of feature extraction with KNN are much better than the traditional DTW algorithm, e.g., the recognition rates of gestures 1, 2, and 3 reached 0.99. Overall, the recognition rates of gestures 4, 5, and 6 are lower than gestures 1, 2, and 3. We hence further look at the false recognition results of the three gestures.

Figure 12 shows that the false recognition results of gestures 4 and 5 are very similar. 86% of the false recognition results of gesture 4 were recognized as gesture 5, and 98% of the false recognition results of gesture 5 were recognized as gesture 4. By analyzing the motion trend of gestures 4 and 5, we found that the difference between gestures 4 and 5 only lies in the direction of rotation, which is weakly reflected in the data of x- and z-axes of gyroscope. The range of motion and the amount of change of different axes of the sensor are not greatly affected by the direction. Therefore, the recognition algorithm of feature extraction with KNN cannot distinguish well the differences between gestures 4 and 5. 84% of the false recognition results for gesture 6 were recognized as gesture 2. This is largely due to similar ranges of motion, i.e., gesture 2 was performed with a small jitter, and gesture 6 with too little range performed by the user was recognized as gesture 2. If the range of the upward fling of gesture 6 is defined with a threshold, then the fatigue of performing gesture 6 would increase and could not be suitable for some people.

Moreover, we deployed the algorithm on Huawei Watch 2 with Snapdragon Wear 2100 processor and tested the algorithm execution time. We used the time module in Java to check the time consumption of feature extraction and the KNN for the test set and then got 12.1 ms and 3.5 ms for every sample. The algorithm's low computing power requirements further contribute to the deployment and research of smartwatch delimiter research on mobile devices.

4.4.3. Impact of Gestures with Different Execution Times. We further discuss the influence of gestures with different execution times on recognition accuracy. Our system can reliably achieve a high recognition rate for gestures with different

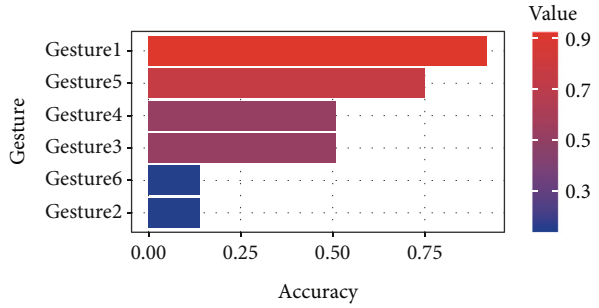


Figure 9: The accuracy of DTW.

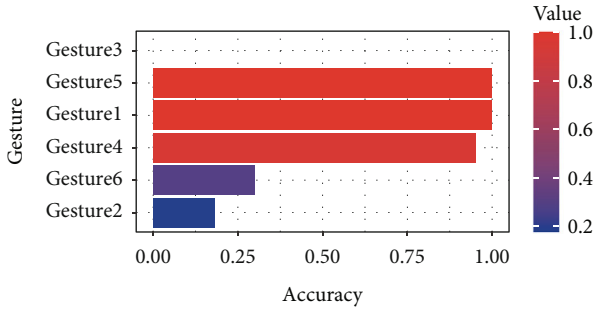


Figure 10: The accuracy of DTW after exclusion of gesture 3.

Table 2: The accuracy of several K.

	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6
$K = 1$	1	0.98	1	0.64	0.47	0.88
$K = 3$	1	0.99	0.99	0.64	0.46	0.79
$K = 5$	0.99	0.99	0.99	0.57	0.41	0.76
$K = 7$	0.99	0.97	0.99	0.42	0.56	0.73

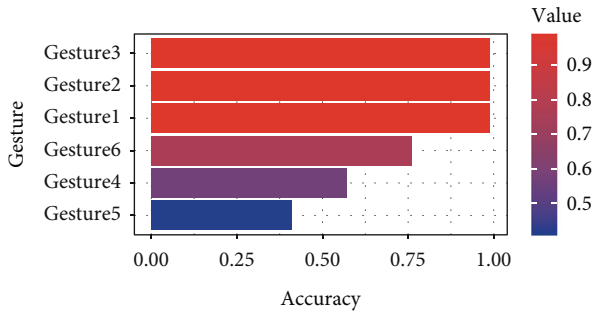


Figure 11: The accuracy of KNN when $K = 5$.

execution times from 0.4 s to 2 s. In the experiment, every participant performed six gestures with as fast as possible, standard, and as slow as possible speed. To test the accuracy of the KNN algorithm for gestures with different execution times, we collected data from another five participants (three males and two females) with the same setting of experiment one. Table 3 shows the accuracy for gestures with different execution times. We observe that the accuracy is similar for gestures with fast and standard execution times, while

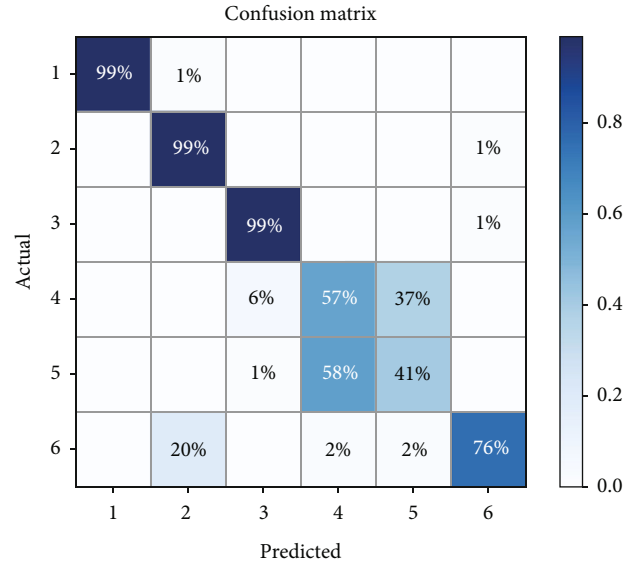


Figure 12: Confusion matrix graph generated using KNN when $K = 5$.

there is a slight decrease in accuracy at shorter execution times. The speed limit of performing the gestures may lead to this difference. The time difference between gestures with fast and standard execution speed is slight (from 0.1 to 0.3 s), leading to similar accuracy. However, the time difference between gestures with slow and standard execution speed is more significant (from 0.4 to 1 s), resulting in slightly lower accuracy. Generally, our system achieves a high recognition rate for gestures with different execution times, benefiting from the five features together.

4.5. Questionnaire Data Analysis

4.5.1. Questionnaire Settings. After completing the experiment task, each participant was asked to fill in a questionnaire to rate the six delimiters on 5-point Likert scales regarding "easy to learn," "easy to perform," "accurate to recognize," "avoid misoperation," and "suitable as the delimiter" (5 for the highest preference and 1 for the lowest preference). We created the questionnaire through an online website and then sent it to each participant through communication applications. Participants fill out the questionnaire via their smartphones or personal computers.

The questionnaire uses a 5-point Likert scale. The 5-point Likert scale has five options with five different scores according to the user's level of agreement, often with scores of 5, 4, 3, 2, and 1. The user chooses suitable options according to their degree of conformity to the declarative statements, and we calculate the total score according to the score assigned to each option of the scale for subsequent analysis. The declarative statements comprise unfavorable and favorable statements. In this experiment, the options indicate five different levels of strongly agree to disagree, and the scores are 5, 4, 3, 2, and 1 if it is a favorable statement and 1, 2, 3, 4, and 5 if it is an unfavorable statement. The gesture with the highest score represents the most

Table 3: Accuracy of feature extraction with KNN for gestures with different execution times.

Execution time	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6
Short	0.95	0.98	1	0.7	0.56	0.77
Standard	1	0.98	1	0.64	0.46	0.79
Long	1	0.88	1	0.6	0.45	0.75

suitable defining gesture from the user's subjective feeling perspective. In designing the questionnaire, it needs to set declarative statements in terms of fatigue, speed of performing gestures, guessability, gesture recognition accuracy, and user's subjective perception, as shown in Table 4.

4.5.2. Experimental Results. We used the chi-square statistics method to calculate the differences of delimiters in the measures. Regarding "easy to learn" and "easy to perform," participants generally regarded that the six delimiters were quite similar. These gestures were simple in form; so, participants thought they were all easy to learn and perform. However, in terms of "accurate to recognize," "avoid misoperation," and "suitable as the delimiter," the scores of gestures 1, 2, and 3 were significantly higher than other gestures (all $p < 0.05$), and there was no significant difference between the three gestures (all $p > 0.05$). In addition, in the questionnaire, participants who regarded gestures 1, 2, and 3 suitable for defining gestures account for 30%, 40%, and 30%, respectively. By combining the above results, the three gestures were selected as the suitable delimiters for further consideration.

Gesture 3 has the highest score, 40% of people thought it was easy to learn, and 60% of people thought it was suitable as a defined gesture. And the recognition rate of gesture 3 was 0.99 by the feature extraction with the KNN recognition algorithm; so, the gesture was considered as the best defined gesture.

The scores of gestures 1 and 2 are high, and the recognition rates of two gestures by the feature extraction with the KNN recognition algorithm are both 0.99. In the questionnaire survey, 20% of people thought that gesture 1 was suitable as the defining gesture, and 60% thought that gesture 2 was suitable as the defining gesture. In a comprehensive view, gesture 1 (shaking wrist right and left) and gesture 2 (shaking wrist up and down) can also be selected as the best defining gestures.

5. Experiment Two: Misoperation Rate of Delimiters

In this experiment, we aimed to investigate misoperation rate of delimiters in representative daily activities. According to experiment one, gesture 1 (shaking wrist left and right), gesture 2 (shaking wrist up and down), and gesture 3 (holding fist and opening) were selected as the candidate delimiters to be tested in this experiment.

5.1. Experimental Settings. The experimental equipment and participants were the same with experiment one.

5.2. Experimental Tasks. We evaluated misoperation rates of the three delimiters in three common scenes in our lives: walking, running, and standing up and sitting down. As a controlled study, we could not cover all daily activities. Instead, we selected three representative activities, that is, walking, running, and standing up and sitting down, to test the misoperation rate of three delimiters. The three scenarios can cover basic day-to-day activities. The data were collected from the participants, who wore the smartwatch to perform ten steps of walking, ten steps of running, and ten times of standing up and sitting down.

As experiment one, we processed the sensor data by the data preprocessing step and then recognized the gesture data to see whether participants accidentally performed gesture 1, 2, and 3 in the three scenes, so as to obtain the misoperation rate of gestures.

5.3. Experimental Results. According to experiment one, the feature extraction with KNN was faster and had better recognition performance than the DTW algorithm. Thus, the feature extraction with KNN was selected as the gesture recognition algorithm in this experiment. The misoperation rates are shown in Table 5.

The misoperation rate of gestures 1 and 2 is 0 in all scenarios. Gesture 3 had 0 in the running and walking scenarios, but 21% in the standing up and sitting down scenario. Therefore, in most scenarios, gestures 1 and 2 are more suitable as the delimiter than gesture 3.

6. Discussion

This study examined six gestures as delimiters for motion gesture interaction with smartwatches. We evaluated the performance of the six delimiters to select the proper ones. First, we used DTW and feature extraction with KNN to obtain gesture recognition accuracy for the delimiters. It is concluded that the feature extraction with KNN has a higher recognition rate for the gesture data, and its recognition rate for gestures 1, 2, and 3 exceeds 0.98. In addition, we checked he misoperation rate of gestures 1, 2, and 3 in three daily scenes. The misoperation rate of the three gestures in the scenarios of walking and running is 0. For standing up and sitting down, the misoperation rate of gestures 1 and 2 is 0, but the misoperation rate of gesture 3 is 21%. Therefore, gesture 1 (shaking wrist left and right) and gesture 2 (shaking wrist up and down) should be more suitable as delimiters for motion gesture interaction on smartwatches. Despite excluding gesture 3 as a delimiter, its excellent recognition accuracy and outstanding performance in the questionnaire still prove its importance as a motion gesture.

Table 4: Average rating of each gesture for each measure.

Measure	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6
Easy to learn	4.3	4.2	4.4	4.2	4.3	4.2
Easy to perform	4.5	4.4	4.4	4.1	4.1	4.4
Accurate to recognize	4.5	4.5	4.1	3.1	3.1	3.3
Avoid misoperation	4.2	4.2	4.1	2.9	3.0	3.3
Suitable as delimiter	4.5	4.4	4.3	3.3	3.2	3.7

Table 5: Misoperation rate of gestures 1, 2, and 3.

Scenario	Gesture 1	Gesture 2	Gesture 3
Running	0	0	0
Walking	0	0	0
Standing up and sitting down	0	0	0.21

We can further improve our work in the following directions. First, for recognition algorithms, only two basic algorithms were used for gesture recognition in this paper. We need to test other algorithms, e.g., recognition algorithms based on the hidden Markov model. Second, for experimental design, this paper only designed six candidate gestures for experiments, and there may be other more suitable defined gestures. Third, participants can be selected from different ages, genders, and occupations. More user data and wider coverage of subjects help to draw more accurate conclusions. Fourth, for the obtained defined gestures, the experiments in this paper were conducted under lab conditions. Considering that the defined gestures are often used together with common action gestures, their practical applications in motion gesture interaction need to be further investigated. Finally, gesture data collection is susceptible to the environment. The sensors that collect the data also produce a certain amount of errors, and even the way the user wears smartwatches could affect collected data. Advances in wearable devices can mitigate the impact of these problems and improve the usability of gesture interaction, and the development of gesture interaction can also promote the progress and popularity of wearable devices.

7. Conclusion

This paper is aimed at selecting suitable gestures as the delimiter for smartwatch motion gesture interaction. To this end, this study firstly selected six candidate gestures (gesture 1: shaking wrist left and right; gesture 2: shaking wrist up and down; gesture 3: holding fist and opening; gesture 4: turning wrist clockwise; gesture 5: turning wrist anticlockwise; gesture 6: shaking wrist up). We conducted two experiments to evaluate the performance of the above six candidate delimiters. We used DTW and feature extraction with KNN to recognize these delimiters. Results showed that gestures 1, 2, and 3 achieved high recognition rate. In the second experiment, during the common scenes in our life, the misoperation rate of gestures 1 and 2 is 0, but the misoperation rate of gesture 3 is 21%. Therefore, gesture 1 and gesture 2 are suitable as motion gesture delimiters for smartwatch interaction.

Data Availability

We collected gesture data from ten participants, including eight men and two women, through sensors in smartwatches.

Additional Points

Research Highlights. (i) A study was conducted to investigate motion gesture delimiters for smartwatch interaction. (ii) "shaking wrist left and right" and "shaking wrist up and down" can serve as delimiters for motion gesture interaction with smartwatches. (iii) Feature extraction with KNN provided higher recognition accuracy than the DTW algorithm. (iv) The study provides insights into designing gesture-based delimiters for smartwatch interaction.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant (2019YFB2102000), in part by the National Natural Science Foundation of China under Grant (62172215), and in part by the Natural Science Foundation of Jiangsu Province (No. BK20200067).

References

[1] S. Pizza, B. Brown, D. McMillan, and A. Lampinen, "Smartwatch in vivo," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 5456–5469, New York, 2016.

[2] D. Pradhan and N. Sujatmiko, *Can smartwatch help users save time by making processes efficient and easier, [M.S. thesis]*, vol. 18, University of Oslo, 2014.

[3] E. Årsand, M. Muzny, M. Bradway, J. Muzik, and G. Hartvigsen, "Performance of the first combined smartwatch and smartphone diabetes diary application study," *Journal of Diabetes Science and Technology*, vol. 9, no. 3, pp. 556–563, 2015.

[4] D. J. Wile, R. Ranawaya, and Z. H. T. Kiss, "Smart watch accelerometry for analysis and diagnosis of tremor," *Journal of Neuroscience Methods*, vol. 230, pp. 1–4, 2014.

[5] L. Heng, "Smartwatch interaction design research," *Technology and Innovation*, vol. 8, p. 73, 2015.

[6] W. J. Hou and W. U. Chun-Jing, "Gestures interaction research based on the data analysis for smart watch," *Packaging Engineering*, vol. 36, no. 22, pp. 13–16, 2015.

[7] C. Xu, P. H. Pathak, and P. Mohapatra, "Finger-writing with smartwatch: a case for finger and hand gesture recognition using smartwatch," in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, pp. 9–14, New York, 2015.

[8] L. Huang, F. Qiaobo, M. He, D. Jiang, and Z. Hao, "Detection algorithm of safety helmet wearing based on deep learning," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 13, article e6234, 2021.

[9] Z. Lu, X. Chen, Q. Li, X. Zhang, and P. A. Zhou, "A hand gesture recognition framework and wearable gesturebased interaction prototype for mobile devices," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 2, pp. 293–299, 2014.

[10] Y. Weng, D. Ying Sun, B. T. Jiang, Y. Liu, J. Yun, and D. Zhou, "Enhancement of real-time grasp detection by cascaded deep convolutional neural networks," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 5, article e5976, 2021.

[11] P. Zhang and Z. S. Liu, "Gesture recognition method based on inertial sensor mpu6050," *Transducer and Microsystem Technologies*, vol. 37, no. 1, pp. 46–53, 2018.

[12] J. Ruiz and Y. Li, "Doubleflip: a motion gesture delimiter for mobile interaction," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2717–2720, New York, 2011.

[13] S.-J. Cho, J. K. Oh, W.-C. Bang et al., "Magic wand: a hand-drawn gesture input device in 3-d space with inertial sensors," in *Ninth International Workshop on Frontiers in Handwriting Recognition*, pp. 106–111, IEEE, Kokubunji, Tokyo, Japan, 2004.

[14] H. Wang and Z. Li, "Accelerometer-based gesture recognition using dynamic time warping and sparse representation," *Multimedia Tools and Applications*, vol. 75, no. 14, pp. 8637–8655, 2016.

[15] P. Hart, "The condensed nearest neighbor rule (corresp)," *IEEE Transactions on Information Theory*, vol. 14, no. 3, pp. 515-516, 1968.

[16] M. Hirota, A. Tsuboi, M. Yokoyama, and M. Yanagisawa, "Gesture recognition of air-tapping and its application to character input in vr space," in *SIGGRAPH Asia 2018 Posters*, pp. 1-2, New York, 2018.

[17] X. Xie and Z. Liu, "Electromyography hand gesture recognition method based on dtw," *Computer Engineering and Applications*, vol. 54, no. 5, pp. 132–137, 2018.

[18] J.-W. Lin and C. Wang, "Backhand: sensing hand gestures via back of the hand," in *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pp. 557–564, New York, 2015.

[19] S. Agarwal, A. Mondal, G. Joshi, and G. Gupta, "Gestglove: a wearable device with gesture based touchless interaction," in *Proceedings of the 8th Augmented Human International Conference*, pp. 1–8, New York, 2017.

[20] M. Baglioni, E. Lecolinet, and Y. Guiard, "Jerktilts: using accelerometers for eight-choice selection on mobile devices," in *Proceedings of the 13th international conference on multimodal interfaces*, pp. 121–128, New York, 2011.

[21] Y. Chen, P. Yang, and X. Chen, "A gesture recognition method based on acceleration feature extraction," *Chinese Journal of Sensors and Actuators*, vol. 25, no. 8, pp. 1073–1078, 2012.

[22] H. Duan, Y. Sun, W. Cheng et al., "Gesture recognition based on multi-modal feature weight," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 5, article e5991, 2021.

[23] X. L. Guo, T. T. Yang, and Y. C. Zhang, "Gesture recognition based on kinect depth information," *Journal of Northeast Dianli University*, vol. 36, pp. 90–94, 2016.

[24] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[25] J.-S. Wang and F.-C. Chuang, "An accelerometer-based digital pen with a trajectory recognition algorithm for handwritten digit and gesture recognition," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 7, pp. 2998–3007, 2012.

[26] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, Eds., "Beyond short snippets: deep networks for video classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4694–4702, Boston, 2015.

[27] D. Ashbrook and T. Starner, "Magic: a motion gesture design tool," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2159–2168, New York, 2010.

[28] J. Ruiz, L. Yang, and E. Lank, "User-defined motion gestures for mobile interaction," in *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 197–206, New York, 2011.

[29] K. Hinckley, J. Pierce, M. Sinclair, and E. Horvitz, "Sensing techniques for mobile interaction," in *Proceedings of the 13th annual ACM symposium on User interface software and technology*, pp. 91–100, New York, 2000.

[30] L. Weberg, T. Brange, and Å. W. Hansson, "A piece of butter on the pda display," in *In CHI'01 Extended Abstracts on Human Factors in Computing Systems*, pp. 435-436, New York, 2001.

[31] K. Katsuragawa, J. R. Wallace, and E. Lank, "Gestural text input using a smartwatch," in *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pp. 220–223, New York, 2016.

[32] J. Liu, Z. Lin, J. Wickramasuriya, and V. Vasudevan, "User evaluation of lightweight user authentication with a single tri-axis accelerometer," in *Proceedings of the 11th International Conference on Human- Computer Interaction with Mobile Devices and Services*, pp. 1–10, New York, 2009.

[33] F. C. Y. Li, D. Dearman, and K. N. Truong, "Virtual shelves: interactions with orientation aware devices," in *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pp. 125–128, New York, 2009.

[34] S. Kratz and M. Back, "Towards accurate automatic segmentation of imu-tracked motion gestures," in *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, pp. 1337–1342, New York, 2015.

[35] J. Gong, X.-D. Yang, and P. Irani, "Wristwhirl: One-handed continuous smartwatch input using wrist gestures," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pp. 861–872, New York, 2016.

[36] F. Kerber, P. Schardt, and M. Löchtefeld, "Wristrotate: a personalized motion gesture delimiter for wristworn devices," in *In Proceedings of the 14th international conference on mobile and ubiquitous multimedia*, pp. 218–222, New York, 2015.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 2774430, 17 pages
https://doi.org/10.1155/2022/2774430

Research Article

MyoTac: Real-Time Recognition of Tactical Sign Language Based on Lightweight Deep Neural Network

Huiyong Li,[1] **Yifan Zhang**,[1] **and Qian Cao**[2,3]

[1]*School of Computer Science and Engineering, Beihang University, Beijing 100191, China*
[2]*School of E-Business and Logistics, Beijing Technology and Business University, Beijing 100048, China*
[3]*National Engineering Laboratory for Agri-Product Quality Traceability, Beijing Technology and Business University, Beijing 100048, China*

Correspondence should be addressed to Qian Cao; caoqian@th.btbu.edu.cn

Received 20 August 2021; Accepted 28 February 2022; Published 25 March 2022

Academic Editor: Hugo Landaluce

Real-time tactical sign language recognition enables communication in a silent environment and outside the visual range, and human-computer interaction (HCI) can also be realized. Although the existing methods have high accuracy, they cannot be conveniently implemented in a portable system due to the complexity of their models. In this paper, we present MyoTac, a user-independent real-time tactical sign language classification system that makes the network lightweight through knowledge distillation, so as to balance between high accuracy and execution efficiency. We design tactical convolutional neural networks (TCNN) and bidirectional long short-term memory (B-LSTM) to capture the spatial and temporal features of the signals, respectively, and extract the soft target with knowledge distillation to compress the scale of the neural network by nearly four times without affecting the accuracy. We evaluate MyoTac on 30 tactical sign language (TSL) words based on data from 38 volunteers, including 25 volunteers collecting offline data and 13 volunteers conducting online tests. When dealing with new users, MyoTac achieves an average accuracy of 92.67% and the average recognition time is 2.81 ms. The obtained results show that our approach outperforms other algorithms proposed in the literature, reducing the real-time recognition time by 84.4% with higher accuracy.

1. Introduction

Gestures are one of the most commonly used ways for humans to convey their expectations [1]. In the transmission of information, gestures often account for a very large proportion. Tactical sign language (TSL) is the gestures utilized in combat operations, using the movements of arms, fingers, and palms to communicate. By recognizing TSL, simple gestures can be applied to deliver rich semantic information to interact with devices, and people can also communicate in a silent environment and beyond the visual range. The traditional gesture recognition is based on computer vision [2, 3], but the situation on the battlefield is so complex that image-based approach is not competent as it may be seriously affected by lighting, shadows, and background. In addition, the required infrastructure is inconvenient for users and not suitable for carrying around. On the contrary, it is more convenient to use the armband with electromyography (EMG) signal acquisition and inertial measurement unit (IMU) to collect the signal data. Among classifiers, deep neural network (DNN) has been widely implemented in gesture recognition [4, 5] due to its hidden layer extract buried features automatically. Because the gestures of tactical sign language are very complex, the system needs to balance both the motion capture accuracy and real-time operating speed. A lightweight model with fewer parameters is essential. However, DNN models for gesture recognition typically stack layers to improve accuracy, resulting in a large model, sacrificing the speed of real-time operation.

In this paper, we present MyoTac, a user-independent high-real-time tactical sign language classification system that makes the network lightweight through knowledge distillation and replacement of the fully connected layer (FCL), so as to balance high accuracy and execution efficiency at the

same time. Knowledge distillation refers to transferring the knowledge in an ensemble of models to a single model [6]. We utilize a portable device, Myo, which enables data transmitting via Bluetooth Low Energy (BLE). The Myo armband acquires 8-channel EMG signals and 9-channel IMU signals, including 3-channel accelerometer signals, 3-channel gyroscope signals, and 3-channel orientation signals. When people wear the Myo armband, our system converts the data received into commands, so that machines can understand human intentions and complete HCI interaction naturally and harmoniously.

Based on the excellent performance of DNN in the fields of image recognition, speech classification, and natural language processing, we develop a hybrid neural network to recognize 30 types of TSL. First, the multimodal mixed data is input into TCNN network to obtain the global characteristics of each channel. Then, the learnt features are input into the B-LSTM to model the temporal features. By training the large model, the knowledge in the large model can be extracted. The soft target obtained from the large model can be combined with the actual label to train the small model. Through the rich information entropy of the soft target, our small model can acquire more knowledge and thus obtain a higher accuracy rate.

In order to evaluate the classification effect of MyoTac, we invited 25 volunteers to test 30 commonly used tactical sign languages and collected 37,500 samples. Volunteers only need to wear the Myo armband on their left arm and repeat the arm and finger movements. By fusing the IMU and EMG signals to comprehensively analyse the fine movements of the arms and fingers, as well as upsampling for fusion, some gestures that cannot be distinguished by a single signal can be more accurately distinguished. When dealing with new users, MyoTac achieves an average accuracy of 92.67%. These results prove the superiority of our algorithm. In order to further reduce the scale of the model and improve the real-time recognition speed, we also compare the large model with the small model obtained through knowledge distillation. The results prove the excellent effect of lightening through knowledge distillation.

The contributions of this paper are summarized as follows:

(1) We propose a multimodal hybrid neural network that combines TCNN and B-LSTM, to realize the classification of user-independent EMG signals of tactical sign language with multichannel correlation and time-varying. The TCNN part extracts spatial features of different channels, and B-LSTM extracts temporal correlation under time series

(2) A novel method based on knowledge distillation is presented to lightweight the TCNN and B-LSTM network, considering the generalization ability of the adopted model can be transferred from a large model to a small model. This method can reduce the space complexity and time complexity of the network, achieving a higher accuracy under the same network scale

(3) We design a nonintrusive real-time tactical sign language recognition system, which can be used for silent human-computer interaction on the battlefield as well as the common sign language recognition (code is available at https://github.com/YifanZhangchn/MyoTac.git). It only requires the participants to wear the Myo armband and carry out military sign language movements without any other pretraining, allowing them to convey instructions to the machine. In addition, we collected a TSL data set with 30 symbol samples and conducted real-time gesture recognition experiments. Experiments on our sign language recognition and network lightweight methods show its excellent recognition performance

The remainder of this paper is organized as follows. We first review related research in Section 2, followed by materials and methods in Section 3, including the data set, signal processing, and the specific deep neural networks. Section 4 presents the results, and Section 5 presents the discussion. Finally, the conclusions of the paper are drawn in Section 6.

2. Related Work

This section reviews the related research in sign language recognition and lightweight neural network.

2.1. Sign Language Recognition. To the best of our knowledge, the classifiers for sign language recognition mainly include traditional machine learning methods [7, 8] and neural network models. In the early work, the researchers extracted handcrafted features and put them into classical machine learning classifiers. Wang et al. extracted shape, depth, and bone trajectory features of the hand to recognize independent gestures through support vector machines (SVM) [9, 10]. Zhuang et al. fused the signal of sEMG and accelerometer on the back of the hand and extracted posture-related features into the linear discriminant analysis (LDA) model to recognize 18 isolated Chinese sign language (CSL) signs [11]. K-nearest neighbour (K-NN) algorithm [12, 13] is also commonly used in recent research. Manually extracting features requires expert domain knowledge, in-depth analysis and heuristic thinking about the problem, as well as combining and experimenting with various refined features, which increases the difficulty of obtaining features with higher matching degrees. The classic machine learning model is a miniature, but it is difficult to construct and select features, so there are problems of low accuracy and weak generalization.

Since the neural network does not need to manually extract features, it can get some hidden features difficult to detect and manual sort, resulting in widely using in sign language recognition [14]. Liu et al. [15] constructed 100 Chinese sign language datasets to capture the motion trajectories of four skeletal joints and input the data into LSTM model. Liang et al. merged multimodal video streams and applied a 3D-CNN model to extract spatial and temporal features in real-time to capture motion information [16].

In addition to analysing gestures through image or video data, biological signals have also been widely used [17–19]. Among them, back-propagation (BP) based neural network, pulse-coupled neural network (PCNN) [20], and probabilistic neural network (PNN) [21] is adopted for the recognition of these complex data. Chen et al. [22] divided the overlapping data segment with a size of 8×52 through a sliding window of 260 ms and applied the continuous wavelet transform (CWT) with a scale of 32 to transform the data into the time-frequency domain. Then, they built a compact CNN model named EMGNet. This model can be used to distinguish static gestures, but it is not enough to infer temporal tactical sign language gestures. The hybrid neural network obtained by combining the recurrent neural network (RNN) and multinetwork model [23] is applied to distinguish the EMG signals acquired from different sign language actions with high accuracy. Zhang et al. [24] presented MyoSign which combines the signals of industrial sensor and EMG sensor. MyoSign proposed a system for inferring American Sign Language and constructed a model integrating multimodel CNN and CTC. However, this kind of portable deep learning system applied to distinguish bioelectric signals usually contains a large number of parameters, which affects its real-time performance.

2.2. Lightweight Neural Network. With the emergence of different types of neural networks, the function of neural networks is becoming more and more powerful, as well as the scale is also expanding. In order to improve the operation efficiency of neural network and deploy it in small embedded devices or mobile devices, the network must be lightweight. At present, the lightweight methods of neural network are mainly reflected in two aspects, namely, lightweight network models and network lightweight methods.

The former is mainly achieved by changing the convolution mode and exchanging information between different convolution layers. The SqueezeNet model of Berkeley Stanford [25] employs the stacking idea of the visual geometry group (VGG) network, emphasizing the application of a 1×1 convolution kernel to compress feature maps. Google's Mobilenet model [26] uses the depth-wise separation filter instead of the traditional convolution method. The Shufflenet model [27] draws on the idea of dividing the convolutional layer into two parts of the MobileNet model, retaining the depth-wise layer to convolve a single channel, and then introducing the shuffle layer to shuffle the channels to ensure the circulation of feature map information in different channels.

The latter is chiefly the method to reduce the space complexity and time complexity of the model by compressing the number and depth of the parameters without obvious influence on the accuracy of the existing network model. Pruning is one of the most commonly used model compression methods [28], and its premise is the overparameterization of deep neural networks. Hao et al. [29] combined the loss with the regulator in training to make weight sparse; then, the importance of the parameters is evaluated according to the absolute value in order to remove the parameters whose importance is lower than the threshold. Quantization is another useful method, which takes the high precision of parameters as the premise. Gupta et al. [30] used two rough methods to realize parameter quantization. One is rounding up nearby, and the other is rounding up or down according to a certain probability.

3. Materials and Methods

3.1. Dataset. To evaluate the performance of the recognition system, we selected 30 tactical sign language gestures commonly used for data collection. The selected 30 gestures (see Table 1) are divided into 5 categories. Twenty-five volunteers (18 males, 7 females, age: 25.2 ± 1.2 yr, height: 173.3 ± 8.1 cm, weight: 64.5 ± 10.8 kg) are recruited in our experiment. Each volunteer wears the Myo armband at the forearm of the left hand (see Figure 1), because TSL is performed with one hand on the left hand. EMG and IMU signals are collected synchronously. The signal data collection consists of raising hands, corresponding tactical sign language movements and arm relaxation movements.

Before data collection, we inform the volunteers of the purpose of the study, the collection procedure, and the duration of collection. During data collection, each subject was required to wear a Myo armband in the same position and in the same orientation of the left hand. They completed the corresponding tactical sign language gestures within 2 seconds and rested for 1 second. Volunteers were required to repeat the above steps 50 times. A total of 37500 samples are collected, and standard Myo software development kit (SDK) is leveraged to collect sensor data.

3.2. MyoTac System Architecture. MyoTac is a user-independent sign language recognition system, which can complete the classification of 30 military sign language commands in real-time. Figure 2 shows the structure of the system. The left side of Figure 2 performs the offline model training. The data is collected by a portable sensor device, Myo, and then transmitted in real-time via BLE. EMG data and IMU data are divided into 2-second segments, and after the data is synchronized by upsampling, they are merged into a 17×400 matrix. The 17 channels correspond to 9 IMU signals plus 8 signals from EMG sensors. 400 is the amount of time series data collected within 2 s at a collection rate of 200 Hz. The formatted data is first put into the TCNN network to extract hidden spatial features. Then, B-LSTM collects context information from multiple different timing modules. In the field of knowledge distillation, soft targets are interpreted as the output obtained after training complex networks, while hard targets refer to the real label of data [31]. Finally, combined with soft target and hard target, the scale of the network is reduced by 86.3%, and the running time is reduced by 33.6% compared with the MyoTac-original model.

The right side of Figure 2 shows the online recognition. We adopt the model that was trained offline to process the new data so we can evaluate the real-time performance of the system. New users do not need to register or perform any other processing, just wear an armband on their left arms, and perform corresponding military sign language actions. The system will detect the short-term energy value of EMG

Table 1: The selected TSL gestures are divided into 5 categories.

Category	Gestures
People	Male, female, commander, hostage, suspect, you, me
Action	Come on, hear, see, advance, message received, hurry up, stop, cover me, not understand, understand, squat down, ignore
Weapon	Pistol, rifle, automatic weapon, shotgun, car
Position	Doorway, corner
Formation	Assemble, single column, two-way column, one-way line

Figure 1: The wearing position of Myo.

data in real-time. When the value rises to the threshold, it will be regarded as a user action, and the corresponding activity segment will be intercepted. The data of EMG and IMU are also merged into a matrix and input into our offline trained model. The system will output the classification results and the time taken for identification in real-time.

3.3. Data Segmentation and Fusion Method of Upsampling. The data obtained from Myo armband has been denoised by filtering and eliminating power frequency interference. The commonly used methods include Butterworth filter to remove high-frequency and low-frequency interference, wavelet denoising, and band stop filtering at 50 Hz to remove power frequency interference.

The EMG signal from Myo and the corresponding short-term energy (see Figure 3) show two similar partial waves when participant performs actions two times. These parts contained in two dotted line boxes are valid data. The time to complete a military sign language action generally does not exceed 2 seconds. Therefore, we set the effective operation time to 2 seconds. During the data collection, our program reminds the volunteers of the beginning and end of each two-second period. Through the analysis of the data of each gesture, it is easily seen that the short-term energy will reach a higher value during the arm activity phase. The short-term energy of EMG signal at time t is defined as

$$E(t) = \sum_{m=t}^{m=t+(T-1)} \left[\sum_{i=0}^{i=7} \mathrm{abs}(S_i(m))w(m-t)\right]^2, \quad (1)$$

where T is the window size, $S_i(m)$ is the EMG time domain signal of i^{th} channel, and $w(t)$ is the window function, which we set as rectangular windows. Therefore, in the recognition part, we set a sliding time window to continuously monitor the short-term energy of the EMG data. When the short-term energy exceeds the threshold, it is judged as the active segment of the signal. In the case that the EMG signal sampling rate is 200 Hz and the number of channels is 8, we obtain an EMG signal matrix $R_{\mathrm{EMG}}^{(8x400)}$ containing the active segment. It can be seen as a picture with a resolution of 8×400.

In the tactical sign language instruction set, there are a numerous number of sign language instructions with the same arm movements and the inconsistent finger movements. For example, both "me" and "female" gestures raise the hand to chest, and likewise "hurry up" and "shotgun" gestures both raise the arm to move up and down. Similarly, there are also some movements where the finger movements are invariable with distinct arm movements. For example, "suspect" and "message received" gestures are both three fingers cocked. "You" and "me" commands are the index finger to straighten and the other fingers to bend. "Pistol" and "rifle" are gestures with index finger and thumb to make a gun. Only when both the IMU signals is applied to distinguish the obvious movements of the arm, and the EMG signals are used to discriminate explicit movements of the fingers, different sign language instructions can be better distinguished. As the sampling rate of the inertial sensor is 50 Hz and the number of channels is 9, there is a data matrix $R_{\mathrm{IMU}}^{(9x100)}$ corresponding to inertial signal. Because of the difference in sampling rate of the inertial sensor and the EMG signal, the upsampling method is adopted. The data matrix $R_{\mathrm{IMU}}^{(9x400)}$ is obtained by linear interpolation of inertial sensor data to ensure the time synchronization of sampling data. At each sampling point, by combining data matrix $R_{\mathrm{EMG}}^{(8x400)}$ and matrix $R_{\mathrm{IMU}}^{(9x400)}$, we get matrix $R^{(17x400)}$ which fused 9 channel inertial sensitive samples and 8 channel EMG samples. Data processing is illustrated in Algorithm 1.

3.4. TCNN Model Construction Based on Correlation between Channels. The traditional method extracts the artificially designed features which refer to the signal features calculated according to the sensor data. They can be independently selected to be input into the machine learning classifier, such as mean absolute value (MAV), root mean square (RMS), zero crossing (ZC), and waveform length (WL). However, it is not always possible to directly find remarkable manual features that can be well generalized to different sensors and users. Based on the superior performance of convolutional neural networks (CNN), it is used to process multimode sensor signals. In order to reflect the timing property (see Section 3.4 for details), we separate the matrix $R^{(17x400)}$ into five clips, and there are 5 data matrices $R^{(17x80)}$.

The muscle signal generated during arm movement can reflect the intensity of muscle activities. As shown in Figure 4, (a) and (b) represent the eight-channel time-

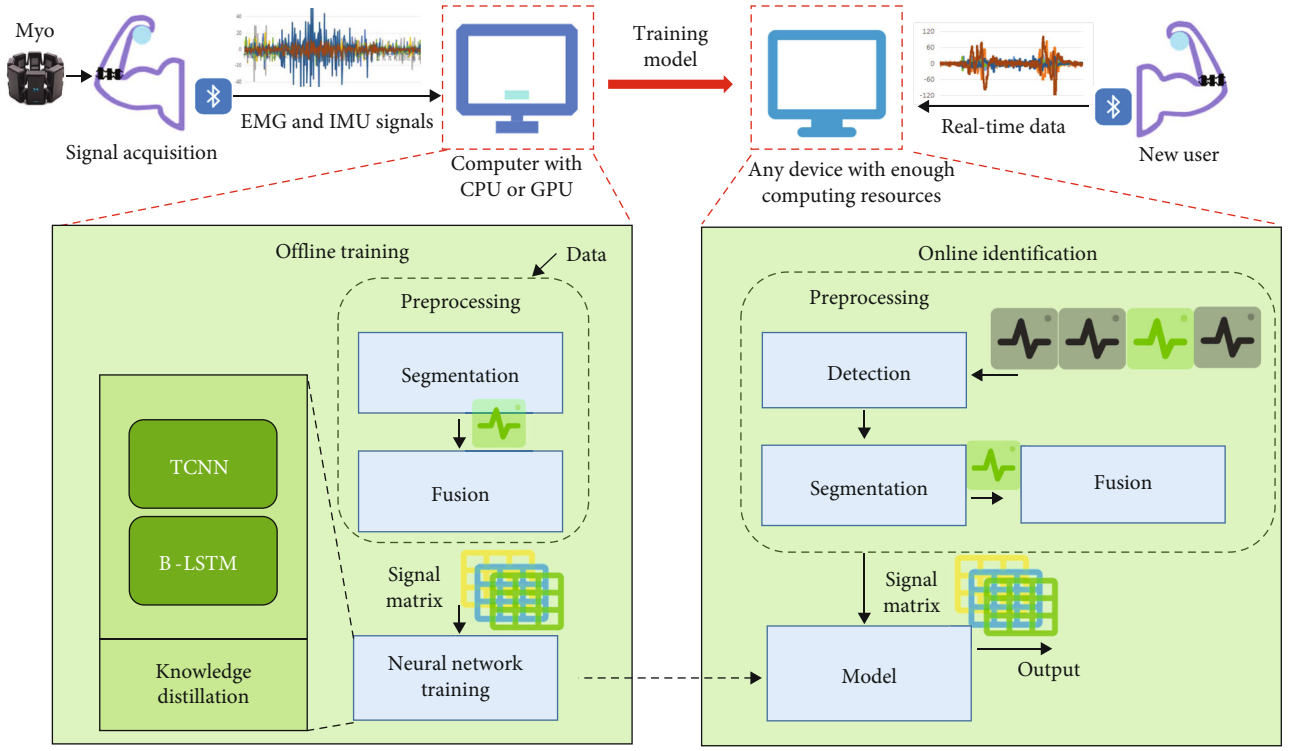


Figure 2: System architecture of MyoTac.

domain signal and frequency-domain signal of two "see" gestures, and (c) and (d) are the signals of "two-way column" gestures. We found that when the gesture "see" is performed, the muscles around channel 4 and channel 7 are more active, and the generated signals are more concentrated in the frequency range of 60-100 Hz. For "two-way column," the muscles near channels 3 and 7 are more active, and the EMG signal collected on channel 3 is also distributed in the low-frequency domain. It is worth noting that because the noise of EMG signal will be caused by power frequency interference at 50 Hz and its harmonics, the built-in algorithm of the Myo armband filters out the power frequency interference, so the distribution of all signals around 50 Hz is attenuated. The EMG signals and frequency domain signals generated by the same gesture in the time domain are very similar, while that of different gestures are quite different.

Figure 5 shows the distribution of the forearm muscles of the human body. When performing tactical sign language, muscle fibres are activated to generate electrical signals, which are transmitted to the surface of the skin. The electrode pads of Myo armband are attached to the surface of the skin, so the signals collected by each electrode pad are the superposition of signals generated by multiple muscle fibres. We analyse the correlation between EMG data channels and find that the Pearson coefficients of adjacent channels are all greater than 0.3, while the Pearson values between nonadjacent channels are not more than 0.2. This is consistent with the fact that the electrical signals captured by adjacent electrode sheets are relatively similar.

In order to master the data relationship between different channels, we design a 3-layer TCNN (see Figure 6). In the first layer, we use the $d \times 3$ convolution filter to obtain the characteristics of the signal matrix. We find that when d is set to 1, the information of each channel is processed separately, and the highest degree of discrimination is achieved. Then, the second layer (1×3) filter is used to learn the high-level representation. After the convolutional layer has extracted the data features, researchers in [32] adopt the FCL to adjust the tensor to 2^n so that it can be input to the next stage of the RNN layer. Due to the large number of parameters in the FCL, the scale of the model will be greatly increased. Therefore, we adjust the size of the feature map to (1×16) through a (17×3) convolutional filter. We apply a maximum pooling layer to each convolutional layer to simplify computational complexity of the network. For all layers, we wield rectified linear unit (RELU) as the activation function. RELU can optimize the gradient dissipation problem in deep neural networks, thereby speeding up the learning speed.

At the end of the TCNN network module, a flattening layer is applied. The flattening layer is used to flatten the output of the previous convolutional layer in order to input features into the B-LSTM module.

3.5. Construction of Bidirectional-LSTM Network Based on Temporal Correlation. Both the EMG signal and the IMU signal describe all the movements during the whole-time sequence. Here, we give the EMG signals of the action "doorway" and "assemble" as examples for illustration. Part

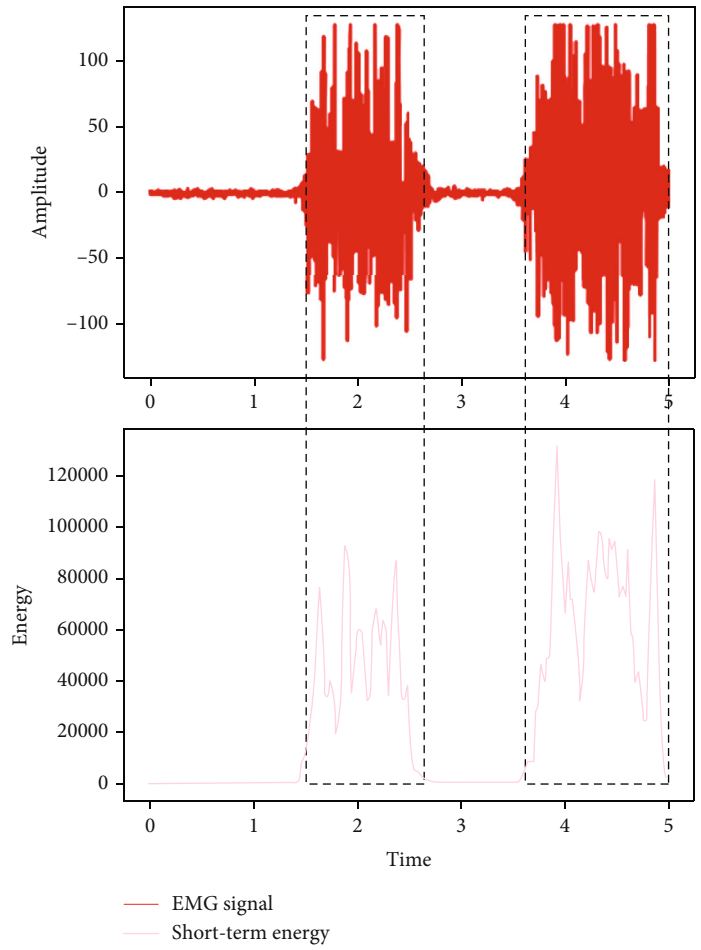


Figure 3: An example of EMG signal and short-term energy.

Input: Real time EMG data $\mathbf{S_{EMG}}$, IMU data $\mathbf{S_{IMU}}$, short time energy threshold **H**
Output: Fused matrix $\boldsymbol{R^{17x400}}$
1: **For** each **t >0.05** s **do**
2: $E(t):=\sum_{m=t}^{m=t+(T-1)}[\sum_{i=0}^{i=7} abs(S_{EMGi}(m))w(m-t)]^2$
3: **If** E(t) > H **then**
4: R_{EMG}^{8x400}:=S_{EMG}(t-0.05:t +1.95)
5: R_{IMU}^{9x400}:=S_{EMG}(t-0.05:t +1.95)
6: R_{IMU}^{9x400}:=Interp (R_{IMU}^{9x400})
7: R^{17x400}:=merge $(R_{EMG}^{8x400}, R_{IMU}^{9x400})$
8: **Return** R^{17x400}
9: **End**
10: **End for**

Algorithm 1: Data processing algorithm.

(a) and part (b) (see Figure 7) describe the actions "doorway" and "assemble," respectively. The expression of "doorway" in TSL is to raise your hand and use your index finger up, left, and then down to draw the shape of a door. When the hand moves in different directions, the most active muscle mass will change, and the signal strength obtained through different channels will change accordingly. In part (a), it is obvious that three active waveforms are corresponding to the signals when the hand moves in three directions. The sign language expression of "assemble" is to raise your hand, extend your index finger to the sky, and turn it twice before retracting. The signal in (b) is divided into four segments, which correspond to the raising of the hand, the first rotation, the second rotation, and the withdrawal. The waveforms of the signals in the middle two segments are similar, and the signals of each channel are sequentially active as they rotate. Due to the high correlation between the signal and time, we divide the signals received into five segments

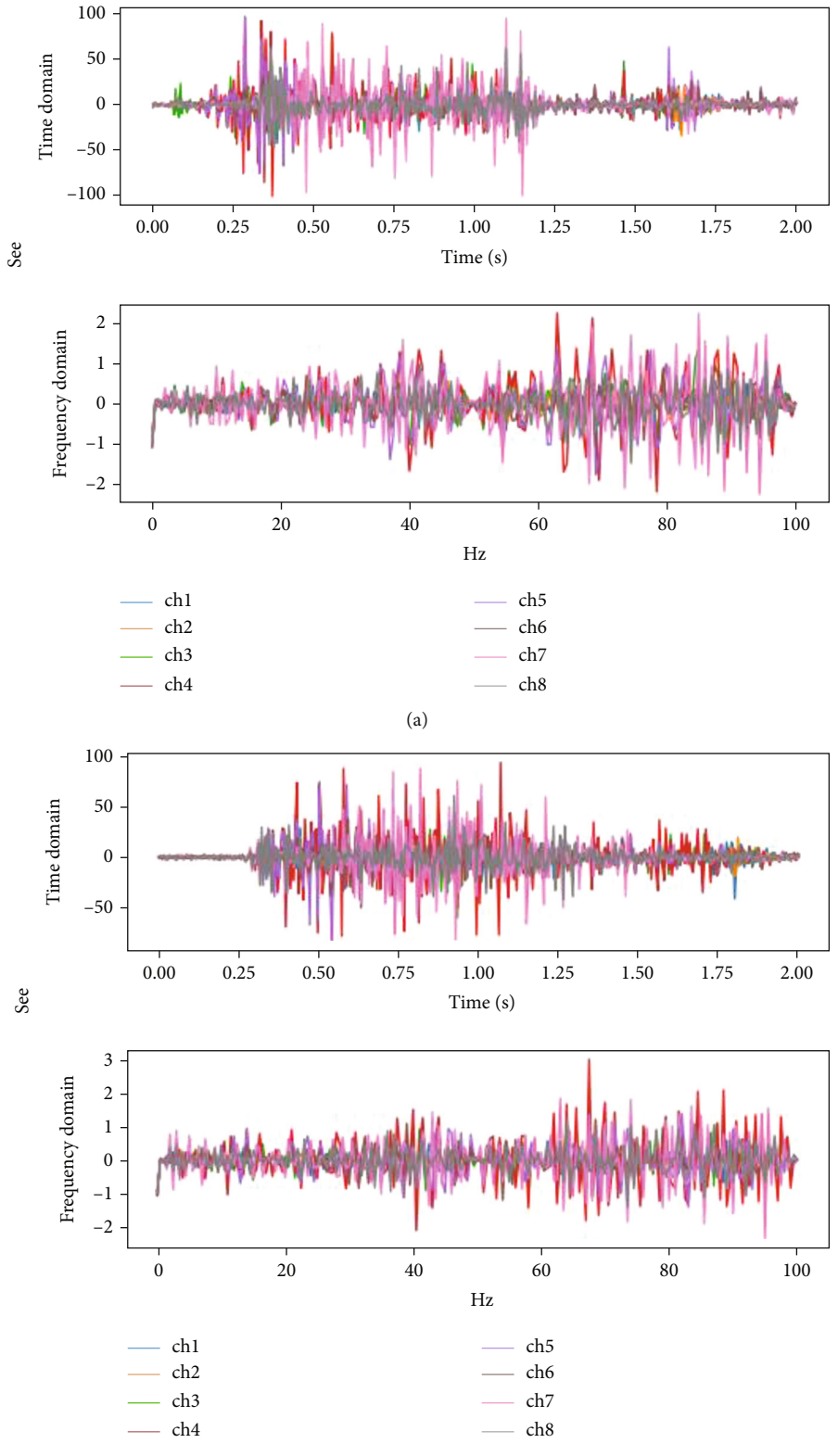


Figure 4: Continued.

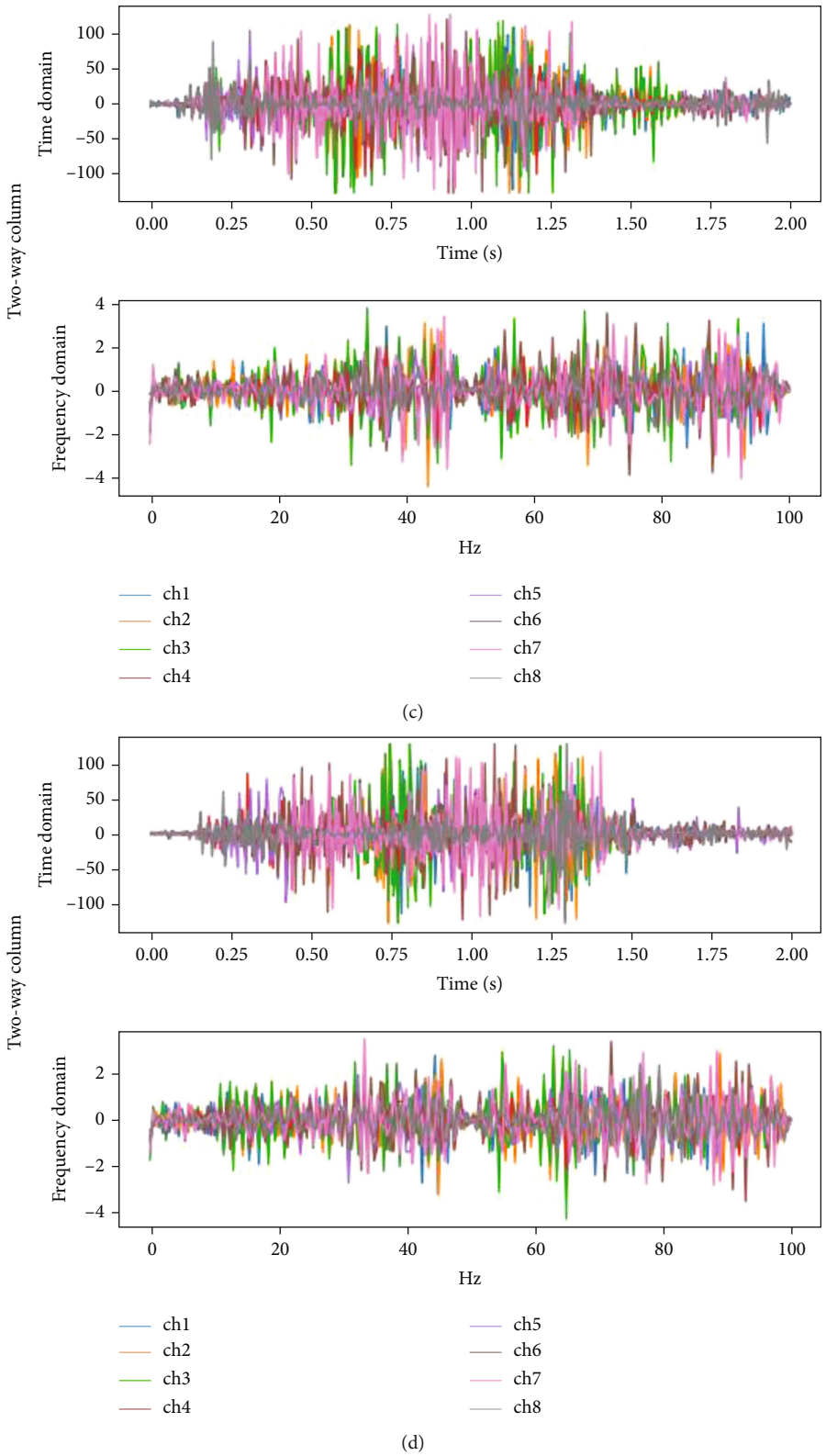


Figure 4: Time domain and frequency domain signals of sample gestures. Ch1-8 corresponds to the signals generated by the eight channels of EMG.

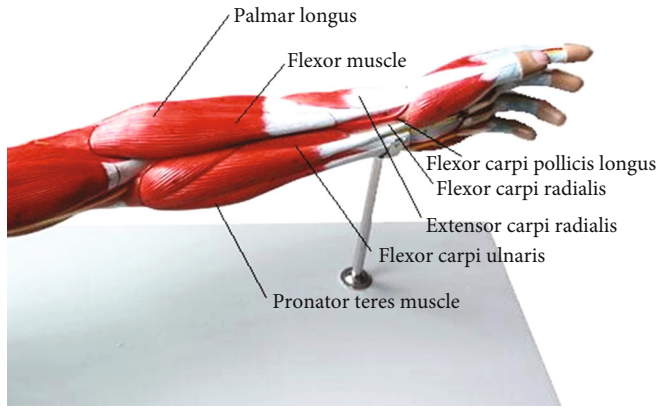


Figure 5: Forearm muscles distribution.

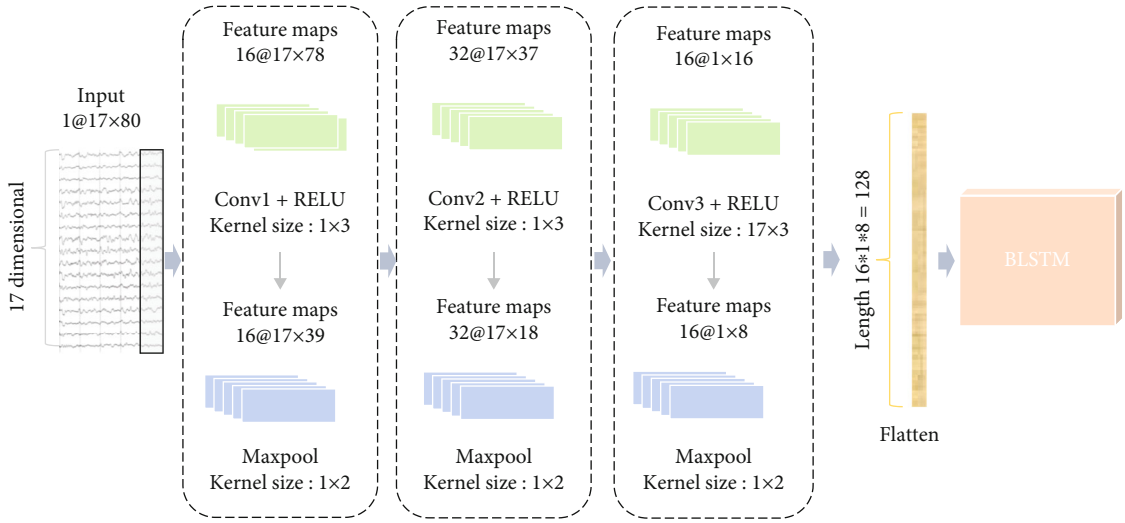


Figure 6: The architecture of TCNN.

for processing in RNN and displayed the dynamic behaviour over time series.

RNN is a neural network capable of processing sequence information. The connections between nodes form a directed graph along the sequence. LSTM, which has input gate, forgotten gate, and update gate to minimize the impact of long-term dependencies, can only be calculated based on the previous information. But many gestures in tactical sign language have the same way of expression at the beginning. In such cases, the inability of LSTM to access future information may cause recognition errors. Therefore, we deployed a B-LSTM as the temporal modelling layer. B-LSTM includes two LSTM layers, which are opposite in the time domain. This makes the output of a certain node depend on both the previous and the subsequent hidden layer state of the sequence at the same time, which ensures that the sign language classification at a certain point in time depends on the entire sequence.

Data needs to be divided into multiple sequences, and the B-LSTM network accepts data of one sequence at a time. We divide the data into 3, 4, 5, 6, 8, and 100 parts, respectively, for experiments, and the model achieve best accuracy when divided into five parts. The action information contained in the data also includes five parts: rest, raise the arm, sign language action, put down the arm, and rest. The specific way proposed to divide it into five parts is to transform the 17×400 data into a $17 \times 5 \times 80$ continuous time series matrix before convolution (see Figure 8). And each convolution only processes one matrix of data at a time. After getting the output of the TCNN network, these time series blocks are recombined as primary data and subsequently input into the bidirectional LSTM network. After the LSTM Layers, we use a dropout layer to reduce overfitting. The dropout rate is adjusted from 0.2 to 0.5 in a parameter selection experiment, and 0.5 is the optimal choice.

Here, the FCL is commonly used to integrate data input into the softmax layer. In addition, in order to avoid the use of the FCL, we try to reshape the output feature map of the B-LSTM to $(1 \times 1 \times 256)$ and turn it into $(1 \times 1 \times 30)$ through a convolution, where 30 is the number of classifications. And then the map feature is fed into the softmax after reshaping. But in fact, the model size of the two methods is almost the same, so the conventional method is still used. Finally, we apply the softmax function to normalize the

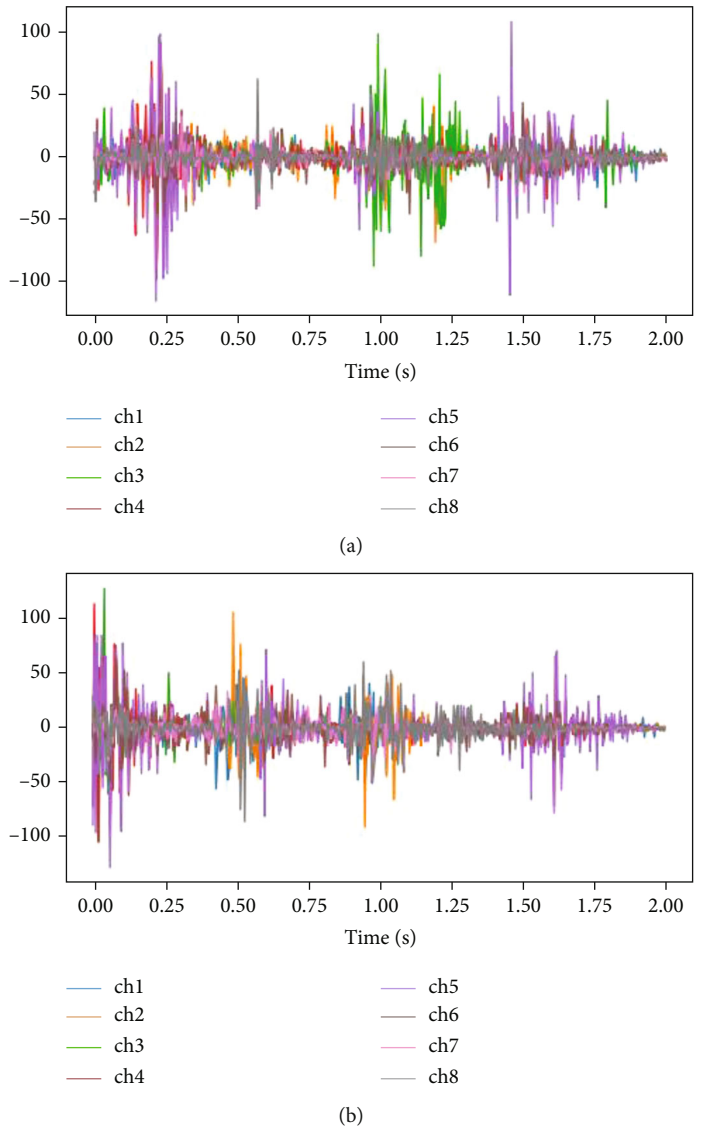


Figure 7: EMG waveforms of "doorway" and "assemble."

output vector, which is then interpreted as the classification probability of the symbol label with index k.

3.6. Lightweight Network through Knowledge Distillation. People intuitively obtain knowledge from fixed real targets, which hinders the abstract view of extracting knowledge, that is, knowledge is a mapping from input vector to output vector [33], which can be extracted from a bulky model set to a small model. The bulky model can be a series of individually trained models or a single but large model. When the cumbersome model is trained, we can deploy another type of training, namely, knowledge distillation, to transfer the knowledge learned from the large model to the student model. Due to the knowledge transfer relationship between the model, the method of knowledge distillation is also called the teacher-student neural network. Especially when recognizing military sign language and the target user has never appeared before, the transferred knowledge has better distinguishing performance because it is liberated from specific instances.

When neural networks conduct classification training, softmax function output s_i is typically applied. The loss function is defined as

$$L(t, y) = -\sum_i t_i \log \frac{\exp\left(s_i/T\right)}{\sum_j \exp\left(s_j/T\right)}, \tag{2}$$

where t_i represents target label. If the real value belongs to category i, t_i equals 1; otherwise, t_i equals 0. T is the distillation intensity that is set to 1 during training and testing, and set to 8 during distillation. A higher T value produces a relatively softer simulation target. We first train a model similar to the final model, but with more convolution kernels and larger parameters. By training this model, we can obtain knowledge about the degree of similarity between actions. After that, we use the same training set to train the small

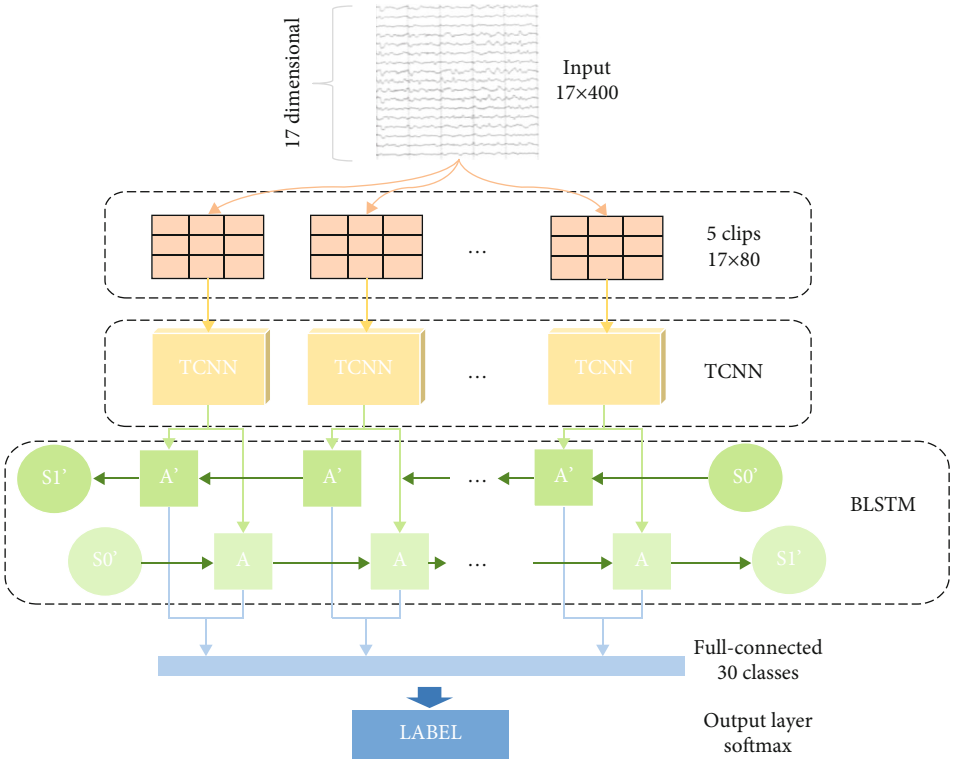


Figure 8: The neural network of MyoTac.

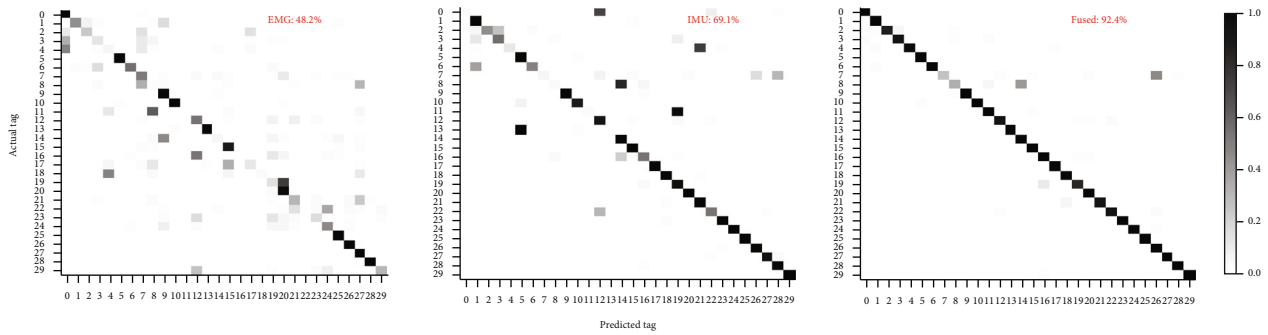


Figure 9: Confusion matrices of the testing results.

model. The logit of the bulky model contains a lot of information and can transfer its generalization ability to the small model, so that the small model has sufficient classification ability. Combined with a distillation intensity value T, the soft target provided by the small model is obtained as shown in Equation (2). The hard target refers to the actual action corresponding to the signal in the training set. Since the information entropy of soft targets is higher, each training instance provides much more information than hard targets. There are two ways to employ soft target and hard target at the same time. The first way is to use real targets to modify the model obtained by soft target training. While the other is to use the weighted average of two objective functions. Considering soft and hard targets in a comprehensive way enables the small model to match the soft target provided by the bulky model while predicting the real target and achieves the best results.

4. Results

4.1. Results of Signal Fused. First, we discuss the importance of fusing the signals of EMG sensors and IMU sensors. In order to fully capture the fine arm and finger movements, we fused the inertial sensor data and the EMG signal data together. Figure 9 shows the confusion matrices, respectively, when only the EMG signals are used, only the IMU signals are used, and the fusion signals are used. The overall recognition accuracy of the thirty gestures in each case was 48.2%, 69.1%, and 92.4%. Table 2 lists the accuracy of tactical sign language in three cases. From Table 2, it is obvious that for some gestures, such as "commander," it is impossible to distinguish when the EMG signal or the IMU signal is adopted alone, but the accuracy is greatly improved when two signals are combined. Therefore, the IMU signal and the EMG signal are complementary to each other for the

Table 2: Accuracy of different signal inputs.

Tag	Gestures	EMG	IMU	Fused signals
0	Male	98.0%	8.0%	99.8%
1	Female	50.0%	98.0%	99.8%
2	Commander	34.0%	50.0%	84.0%
3	Hostage	22.0%	60.0%	90.0%
4	Suspect	12.0%	20.0%	94.0%
5	You	99.8%	98.0%	99.8%
6	Me	60.0%	54.0%	96.0%
7	Come on	56.0%	12.0%	36.0%
8	Hear	4.0%	8.0%	42.0%
9	See	98.0%	98.0%	98.0%
10	Advance	96.0%	86.0%	96.0%
11	Message received	0.0%	6.0%	98.0%
12	Hurry up	58.0%	88.0%	90.0%
13	Stop	96.0%	0.0%	99.8%
14	Cover me	12.0%	99.8%	99.8%
15	Not understand	86.0%	99.8%	99.8%
16	Understand	0.0%	58.0%	96.0%
17	Squat down	20.0%	1.0%	99.8%
18	Ignore	8.0%	98.0%	99.8%
19	Pistol	24.0%	92.0%	82.0%
20	Rifle	94.0%	99.8%	99.8%
21	Automatic weapon	4.0%	98.0%	88.0%
22	Shotgun	0.0%	58.0%	90.0%
23	Car	26.0%	96.0%	99.8%
24	Doorway	52.0%	99.8%	98.0%
25	Corner	99.8%	99.8%	99.8%
26	Assemble	98.0%	98.0%	99.8%
27	Single column	1.0%	90.0%	99.8%
28	Two-way column	98.0%	98.0%	99.8%
29	One-way line	40.0%	99.8%	98.0%

measurement of muscle activities and the evaluation of arm movements, realizing the accurate recognition of sign language instructions.

Zhang et al. [24] apply multi-CNN networks to extract the features of the accelerometer, gyroscope, orientation, and EMG sensors and then merge the resulting tensor to carry out the information interaction between the modals, which is also a signal fusion method. According to this method, we establish a three-layer CNN for each sensor and merge the output of multi-CNN into a multichannel tensor. We pass multichannel tensor through the convolutional layer, flattening layer, and dropout layer and inputted it into the LSTM network to extract temporal features. Taking the data of volunteer 25 as the test set, the comparison between above method and our method is shown in Figure 10. It can be seen that the best accuracy of a single convolutional network (our) with the upsampling method is 92.67%, higher than multi-CNN. It is mainly because the two types of data after upsampling are well synchronized, and the characteristics are not strongly correlated in each channel.

4.2. Comparison of Classification Systems. We use leave-one-user-out cross-validation to evaluate the recognition accuracy of MyoTac in 30 TSL gestures, as shown in Figure 11. The recognition accuracy of no. 1 and no. 21 volunteers is relatively high, because their movements and postures are accurate. No. 8 volunteers' recognition accuracy rate is slightly lower than that of others, since her arm is too thin to keep the armband in the same position. In addition, volunteer no. 11 has a large body weight and a high amount of fat, resulting in a relatively low accuracy. Mainly speaking, the average accuracy of 25 volunteers is 92.4%, and the standard deviation is 2.3%. Different volunteers not only have differences in the outer shape of the arm, including fat thickness and arm circumference, but also different behaviour habits, including the power of the action and some subtle movement differences. But MyoTac can still achieve satisfactory results. Moreover, the recognition accuracy of volunteers is above 88%. This illustrates the generalization ability of MyoTac across users.

In order to further prove the accuracy and high real-time performance, our model is compared with several state-of-the-art researches [22, 24]. Researchers have applied deep learning methods to the field of EMG signal and gesture recognition and have explored several effective network frameworks.

EMGNet [22] builds a compact CNN model to process the time-frequency data and get the classification results. We choose it for comparison as it also reduces the weight of gesture classification model base on EMG signal. MyoSign [24] is an American Sign Language recognition system which inputs the processed data into a complicated model. In order to realize end-to-end sign language recognition, they added a 10-width connectionist temporal classification (CTC) beam decoder at the end of the model.

These two models are compared with MyoTac in the MyoDataSet [34] and the tactical sign language instruction data set we collected. MyoDataSet is a seven-category data set collected by Myo, including seven gestures: neutral, hand close, wrist extension, ulnar deviation, hand open, wrist flexion, and radial deviation. In the first case, after shuffle the MyoDataSet, the training set and the test set are divided at a ratio of 7/3. Then, in the second case, the data of two volunteers of different genders in MyoDataSet are used as test set, while the data of others are used as training set. There is also user-independent data set division on our tactical sign language instruction set data. For MyoSign proposed by Zhang et al., since the data set and model-related parameters are not disclosed in this paper, we use our tactical sign language instruction set to perform the same training set and test set division to compare the two models. In terms of parameters, including training batches, total epochs, and learning rate, both models use the same values. The comparison results are shown in Table 3, and the accuracy of the specific iterative process of the three models when using our military sign language instruction data set is shown in Figure 12.

From Table 3, it can be found that our model has a higher accuracy than other methods. The gestures collected by MyoDataSet remain static during the collection process,

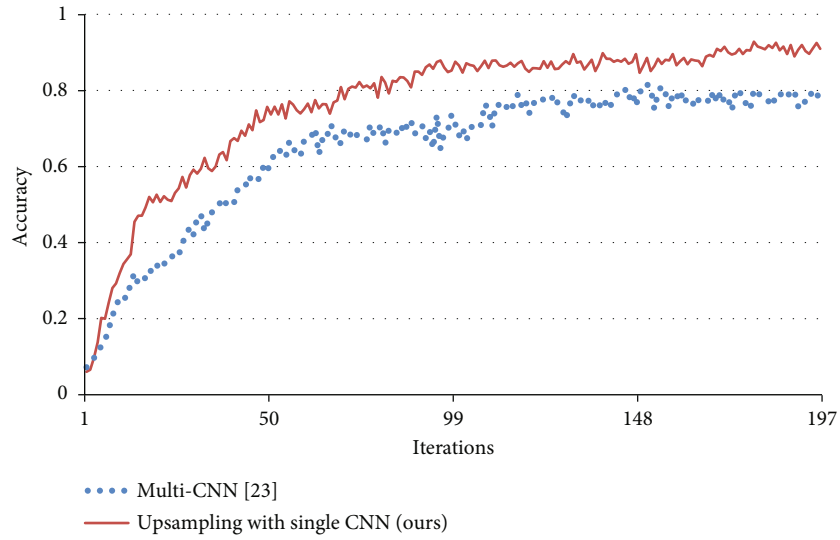


Figure 10: Accuracy of different confusing methods.

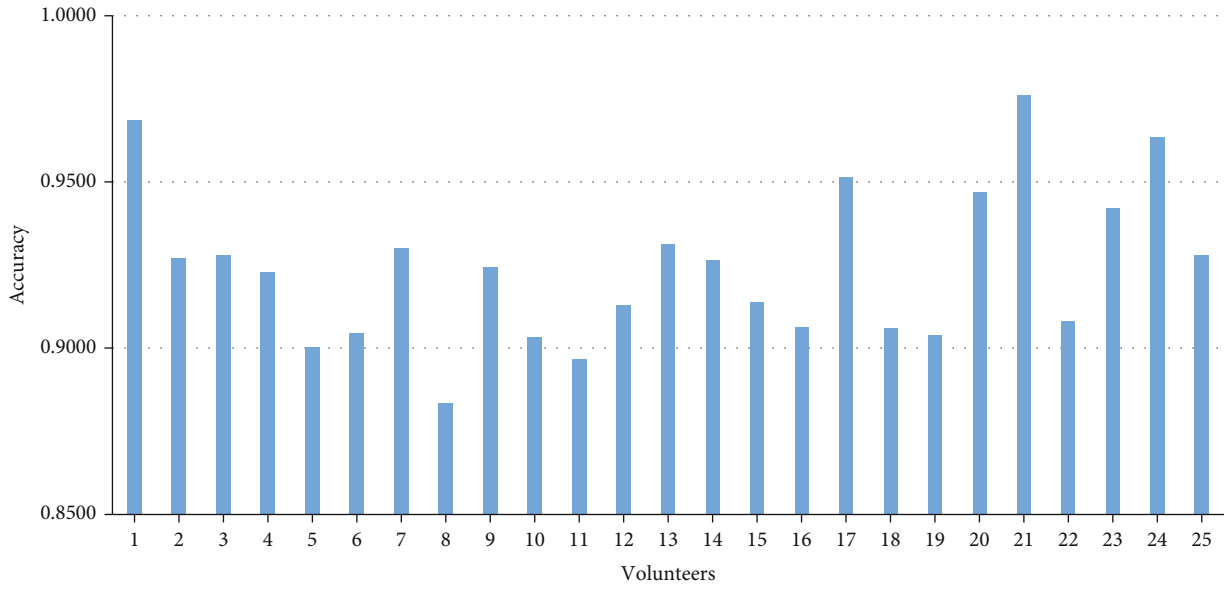


Figure 11: Recognition accuracy across volunteers.

Table 3: Comparison results of 3 models.

Model	Accuracy (MyoDataSet)	Accuracy (MyoDataSet + user independent)	Accuracy (our dataset)	Model size	Processing time (our dataset)
EMGNet	94.3%	82%	13.1%	744 k	—
MyoSign	—	—	91.3%	17.0 M	18 ms
MyoTac	98.1%	85%	92.4%	3.7 M	2.37 ms

while the tactical sign language commands correspond to changing movements. It can be seen from Table 3 that although the model of EMGNet is small, it sacrifices the ability to process time-varying data containing timing information, and the accuracy of distinguishing tactical sign language is only slightly greater than random. Our method can handle both dynamic data and static data splendidly. The experimental results prove the necessity of combining B-LSTM to establish a time dynamic model. Compared with MyoSign, we find that our model reduces the model parameters through knowledge distillation and replacement of the FCL, while maintaining a high accuracy. MyoSign uses 3D convolution, which has one more dimension, and the total amount of calculation to obtain the output of this layer is

Figure 12: Accuracy of three models using our dataset.

also one more dimension than 2D convolution, which slows down the calculation speed of the model. Moreover, CTC is usually applied to the scene where the fixed-length sequence is converted to the variable-length sequence. It is not completely applicable to the sign language classification problem where the output is fixed-length and further reduces the processing time.

4.3. Results of Knowledge Distillation. In this section, we evaluate the effect of knowledge distillation on the lightweight of hybrid networks. First, we train a single cumbersome network to complete the recognition model and then use the distillation intensity value T to convert the output of the model into a soft target with rich information entropy, which is provided to the small model for learning. Table 4 compares several different ways of training small models. Here, we use the data of one volunteer with the highest accuracy in cross training as the test set, and the data of other volunteers as the training set. We first evaluate the results of using only soft targets and get an accuracy of 94.2%, followed by the assessment of applying hard labels with accuracy of 95.6%. Then, we apply both the hard target and soft target, including the weighted average of both targets and hard target correction which is to train the model with the soft target and to revise with the following hard target. The accuracy of the two methods is 97.1% and 97.0%, respectively.

From Table 4, it can be concluded that the application of soft targets alone for training has the worst effect, followed by the use of hard targets alone. Combining soft targets and hard targets to train small models can achieve better training effects. The highest accuracy rate is obtained by the weighted average of soft targets and hard targets, in which we use a relative weight of 0.8 on the cross-entropy for the hard targets. It is considered that soft targets define a rich similarity structure over the gesture, that is, they contain rich information of different gestures which are more similar and whose differences are greater, so the small model can learn knowledge quickly during training. It is similar to that students can start learning expeditiously under the guidance of a teacher. However, due to the large gap between the soft target and the real label, the complete application of the soft target for training leads to a large deviation. Just like a student who only listens to the teacher's teaching without self-study, he cannot succeed. When soft targets and hard targets are combined for training, the small model has both soft targets to provide rich knowledge and hard targets to provide accurate classification. Similar to students who combine teacher guidance and their own diligence, they can reach the highest level of knowledge.

Table 4: Accuracy of training small models in different ways.

Algorithm	Accuracy
Soft target	0.942
Hard target	0.962
Weighted average of soft and hard targets	0.971
Hard target correction	0.970

Table 5: Comparison between original model and small model.

Algorithm	Runtime (ms)	Parameter (M)
MyoTac-original	3.57	27.0
MyoTac-small	2.62	7.1
MyoTac-without FCL	2.37	3.7

We evaluate the runtime and model size of 3 models (see Table 5): the original MyoTac model, the model after knowledge distillation, and the model without FCL. Volunteers can only make one sign language action, so the inference model only needs to process and classify one original test data in a certain period of time. The parameter of the smallest model is only 3.7 M, and the average running time of all

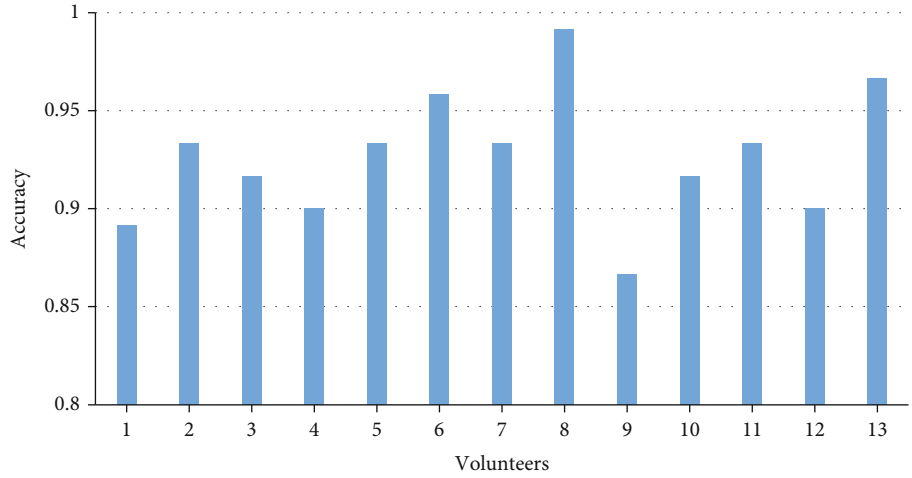


Figure 13: Real-time accuracy across volunteers.

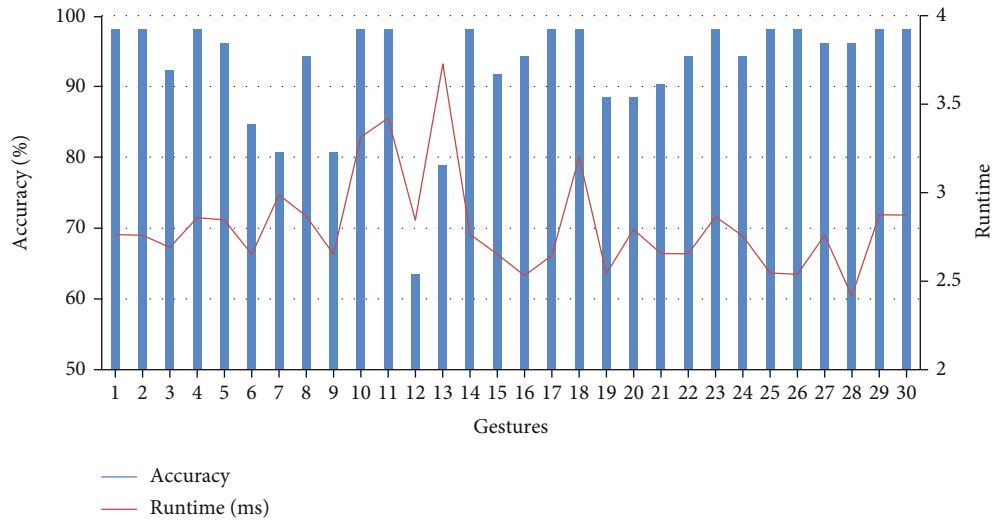


Figure 14: Real-time accuracy and real-time running time of 30 tactical sign language commands.

gesture is 2.37 ms. Without affecting the accuracy, it has faster computing speed and smaller storage scale than before.

4.4. Results of Real-Time Recognition. In order to achieve natural human-computer interaction, MyoTac must be able to complete sign language inference in real-time. The capacitance of the lithium battery used in Myo armband is 220 mAh, and the average measured current during operation is about 9 mA. Therefore, the battery duration of the system is about 24.4 hours. We verified the online operation with 13 participants who were new users to our system. Before the experiment, participants were asked to learn the tactical sign language gestures to avoid the influence of nonstandard actions on the recognition results. During the experiment, participants were asked to select one of the thirty tactile sign language pictures without repetition until all 30 gestures were tested. Each participant performs the above process four times.

The experimental results are shown in Figures 13 and 14. The overall average accuracy of real-time military sign language inference is 92.67%, and the average runtime is 2.81 ms. For 53.3% gestures in TSL, the accuracy of inference is higher than 95% in this experiment. The accuracy of all gestures except "message received" is higher than 75%. The "message received" gesture has the largest number of incorrect judgements, and the accuracy rate is only 63.46%. This is because the arm movement trajectory of "message received" is the same as that of "heard," "pistol," and "two-way column," which leads to that finger movements have to be involved in the inference. However, the finger movements of "message received" and "hearing" are also awfully similar. The signals recognized by the eight-channel Myo armband are not enough to distinguish the extremely similar finger movements under the influence of larger movements.

5. Discussion

This study proposes a lightweight hybrid neural network that achieves tactical sign language classification by using the multimodal data of EMG and IMU. When the user is

new to our system, the accuracy of real-time classification will not be reduced. This study proves that neural network has high redundancy, and the parameters and network layers can still be reduced while achieving high accuracy. Moreover, the reasonable neural network structure which is combined with data can achieve better classification performance under the same numbers of parameters. In the network lightweight, the method of combining soft target with hard target in knowledge distillation is adopted. It also proves that the soft target which obtained by training the large model in knowledge distillation can be used to guide the training of small models and can be applied to the mixed neural network of the convolution layer and BLSTM layer.

The results of this study also show that the increase of data is beneficial to the accuracy of the algorithm. The amount of data in the test set is changed to test the recognition of the same subject. The result is that with the increase of training set data, the accuracy of the action is positively correlated. This is because the key information of gesture movement is extracted, and the difference of movements among different groups can also be recognized by the network. At the same time, the importance of standardizing the acquisition signal is also obvious. When collecting the first experimental data set, the way different participants wore Myo armbands was not specified in detail. This variable greatly affects the classification results of data, so that we refined some criteria of data collection and rebuilt the second data set.

Section 4.2 discusses the necessity of signal fusion. The EMG signal obtained at 200 Hz sampling rate must be fused with IMU signal to achieve better classification effect. However, it is possible to use only the EMG signals for classification. Since the EMG signals can distinguish more precise finger movements, there should be a certain degree of division for arm movements. One of the reasons we need to fuse IMU signal is that the frequency of useful signal of EMG signal is 0-500 Hz, while the sampling rate of Myo is only 200 Hz.

The experiments in terms of network compression are not sufficient. We will conduct further research on training methods that combine soft and hard targets. It is planned to evaluate the accuracy of the step-by-step training method and train each stage separately without affecting the weight of the second-stage network. For the distillation intensity value T for distilling out the soft target, the optimal value was not obtained. In addition, due to the particularity of the convolutional network used, methods such as depthwise convolution and channel pruning can also be mixed for further network compression.

Although the experimental results are satisfactory, there still exists limitation of MyoTac. For example, with regard to some people with thin arms, the installation is unable to be fixed on the arm stably, resulting in unstable signals. Second, the new gesture cannot be dynamically adapted. In the future, we will struggle to change the classification model to form a miniaturized adaptive classification system. In addition, we will design a multimodal sensor suitable for various arm thickness to make the system more robust. In the application of sign language recognition, the volunteers sat to collect data. We also analysed the standing situation and found that it has little effect on the analysis results. For the application scenario of walking, because the signal of inertial sensor will be distorted by movement, it may need a separate sensor to measure the influence of walking and correct the signal data.

6. Conclusions

We present MyoTac, a sign language recognition system based on EMG, which uses EMG and IMU multimodal signal fusion to classify sign language through a knowledge distillate lightweight neural network. We collected the standard tactical sign language data from 25 volunteers and completed a multimodal data set. A convolution combined B-LSTM hybrid network is designed, as well as the reduction for scale of the network by knowledge distillation and less use of FCL. Our system can effectively distinguish different sign languages on the premise of user independence. The average accuracy of real-time classification inference is 92.67%, and the average real-time running time is 2.81 ms. The encouraging performance of MyoTac proves its potential for silent human-computer interaction applications.

Data Availability

Data is available at https://github.com/YifanZhangchn/MyoTac.git.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

The authors would like to thank the collaboration of all volunteers who participated in data collection. This research was funded by National Natural Science Foundation of China (no. 61702018).

References

[1] X. Liu, J. Sacks, M. Zhang, A. G. Richardson, T. H. Lucas, and J. Van der Spiegel, "The virtual trackpad: an electromyography-based, wireless, real-time, low-power, embedded hand-gesture-recognition system using an event-driven artificial neural network," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 64, no. 11, pp. 1257–1261, 2017.

[2] A. Ghulam, "A kinect-based sign language hand gesture recognition system for hearing- and speech-impaired: a pilot study of Pakistani sign language," *Assistive Technology*, vol. 27, no. 1, pp. 34–43, 2015.

[3] J. S. Artal-Sevil and J. L. Montanes, "Development of a robotic arm and implementation of a control strategy for gesture recognition through leap motion device," in *2016 Technologies Applied to Electronics Teaching (TAEE),*, Seville, Spain, 2016.

[4] A. Khanna and S. A. Muthukumaraswamy, "Cost-effective system for the classification of muscular intent using surface electromyography and artificial neural networks," in *2017*

International conference of Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 2017.

[5] W. Zhang and J. Wang, "Dynamic hand gesture recognition based on 3D convolutional neural network models," in *2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC)*, Banff, AB, Canada, 2019.

[6] A. Mishra and D. Marr, *Apprentice: Using Knowledge Distillation Techniques to Improve Low-Precision Network Accuracy*, 2017, https://arxiv.org/abs/1711.05852.

[7] C. P. Robinson, B. Li, Q. Meng, and M. T. Pain, "Pattern classification of hand movements using time domain features of electromyography," in *Proceedings of the 4th International Conference on Movement Computing*, London, United Kingdom, 2017.

[8] B. Bauer and H. Hienz, "Relevant features for video-based continuous sign language recognition," in *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 64–75, Grenoble, France, 2000.

[9] H. Wang, X. Chai, X. Hong, G. Zhao, and X. Chen, "Isolated sign language recognition with Grassmann covariance matrices," *ACM Transactions on Accessible Computing (TACCESS)*, vol. 8, no. 4, p. 14, 2016.

[10] M. Perusquía-Hernández, S. Ayabe-Kanamura, K. Suzuki, and S. Kumano, "The invisible potential of facial electromyography: a comparison of EMG and computer vision when distinguishing posed from spontaneous smile," in *CHI '19: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, Glasgow, Scotland, UK, 2019.

[11] Y. Zhuang, B. Lv, X. Sheng, and X. Zhu, "Towards Chinese sign language recognition using surface electromyography and accelerometers," in *2017 24th International Conference on Mechatronics and Machine Vision in Practice (M2VIP),*, Auckland, New Zealand, 2017.

[12] Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[13] M. E. Benalcázar, A. G. Jaramillo, A. Zea, A. Páez, and V. H. Andaluz, "Hand gesture recognition using machine learning and the myo armband," in *2017 25th European Signal Processing Conference (EUSIPCO)*, pp. 1040–1044, Kos, Greece, 2017.

[14] S. Yang and Q. Zhu, "Video-based Chinese sign language recognition using convolutional neural network," in *2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN),*, pp. 929–934, Guangzhou, China, 2017.

[15] T. Liu, W. Zhou, and H. Li, "Sign language recognition with long short-term memory," in *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 2871–2875, Phoenix, AZ, USA, 2016.

[16] Z. Liang, S.-B. Liao, and B.-Z. Hu, "3D convolutional neural networks for dynamic sign language recognition," *The Computer Journal*, vol. 61, no. 11, pp. 1724–1736, 2018.

[17] V. Becker, P. Oldrati, L. Barrios, and G. Sörös, "Touchsense: classifying¨ finger touches and measuring their force with an electromyography armband," in *Proceedings of the 2018 ACM International Symposium on Wearable Computers*, pp. 1–8, Singapore, 2018.

[18] K. M. Sagayam, A. D. Andrushia, A. Ghosh, O. Deperlioglu, and A. A. Elngar, "Recognition of hand gesture image using deep convolutional neural network," *International Journal of Image and Graphics*, vol. 1, p. 2140008, 2021.

[19] E. Rahimian, S. Zabihi, S. F. Atashzar, A. Asif, and A. Mohammadi, "Surface EMG-based hand gesture recognition via hybrid and dilated deep neural network architectures for neurorobotic prostheses," *Journal of Medical Robotics Research*, vol. 5, p. 2041001, 2020.

[20] R. Forgac and I. Mokris, "Feature generation improving by optimized PCNN," in *Applied Machine Intelligence and Informatics SAMI 2008. 6th International Symposium on Volume*, pp. 203–207, Herlany, Slovakia, 2008.

[21] B. Nan, T. Hamamoto, and T. Tsuji, "FPGA implementation of a probabilistic neural network for a bioelectric human interface," in *Circuits and Systems, MWSCAS '04*, vol. 3, pp. 29–32, Hiroshima, Japan, 2014.

[22] L. Chen, L. Fu, Y. Wu, H. Li, and B. Zheng, "Hand gesture recognition using compact CNN via surface electromyography signals," *Sensors*, vol. 20, no. 3, p. 672, 2020.

[23] J. Machado, M. C. Tosin, L. B. Bagesteiro, and A. Balbinot, "Recurrent neural network for contaminant type detector in surface electromyography signals," in *42nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) in conjunction with the 43rd Annual Conference of the Canadian Medical and Biological Engineering Society*, Montreal, QC, Canada, 2020.

[24] Q. Zhang, D. Wang, R. Zhao, and Y. Y. MyoSign, "MyoSign: enabling end-to-end sign language recognition with wearables," in *the 24th International Conference*, Marina del Ray, California, March 2019.

[25] A. G. Howard, M. Zhu, B. Chen et al., "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," 2019, https://arxiv.org/abs/1602.07360.

[26] G. H. Andrew, Z. Menglong, C. Bo et al., *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*, 2017.

[27] X. Zhang, X. Zhu, M. Lin, and J. Sun, *ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices*, 2017.

[28] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," 2016, https://arxiv.org/abs/1611.06440.

[29] L. Hao, K. Asim, and D. Igor, "Pruning filters for efficient ConvNets," *International Conference on Learning Representations (ICLR)*, 2017.

[30] S. Gupta, A. Agrawal, and K. Gopalakrishnan, "Deep learning with limited numerical precision," in *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, 2015.

[31] G. Hinton, J. Dean, and O. Vinyals, "Distilling the knowledge in a neural network," in *28th Conference on Neural Information Processing Systems*, Montreal, Canada, 2014.

[32] D. Wu, H. Li, X. Liu et al., "Design of gesture recognition system based on multi-channel myoelectricity correlation," in *2019 IEEE Global Communications Conference (GLOBECOM)*, Waikoloa, HI, USA, 2019.

[33] R. G. Lopes, S. Fenu, and T. Starner, *Data-Free Knowledge Distillation for Deep Neural Networks*, 2017, https://arxiv.org/abs/1710.07535.

[34] U. Côté-Allard, C. L. Fall, A. Drouin et al., "Deep learning for electromyographic hand gesture signal classification using transfer learning," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 4, pp. 760–771, 2019.