杭州電子科技大学

硕 士 学 位 论 文

题　　目：智能手表周边手势设计与识别

研 究 生　张博洋

专　　业　计算机科学与技术

指导教师　徐海涛 教授　木下 雄一朗 教授

完成日期　（　　　　　　　　）

杭州电子科技大学硕士学位论文

智能手表周边手势设计与识别

研 究 生： 张博洋

指导教师： 徐海涛 副教授 木下 雄一朗 副教授

2024 年 5 月

DESIGN AND RECOGNITION OF AROUND-SMARTWATCH GESTURES

BY

BOYANG ZHANG

THESIS

presented to the faculty of

the Graduate School of Hangzhou Dianzi University

in partial fulfillment of the requirements

for the degree of Master of Science

Under the Supervision of Professor Haitao Xu and Professor Kinoshita Yoichiro

05, 2024

摘　要

可穿戴设备给人们的日常生活带来更丰富，更便捷的体验。新式设备能提供更多的多媒体功能，但对于用户来说，智能设备的穿戴体验并不局限于新的多媒体功能，还包含了操作舒适性、便捷性等其他人机交互（Human-Computer Interaction, HCI）因素。传统的智能设备通过屏幕与用户交互，而传感器技术的发展使智能可穿戴设备周边的手势交互方式也逐渐成为可能。过去的学者也有深耕于此的研究，但碍于算法和传感器精度等多方面原因，学者们通常无法兼顾技术的实用性和手势集的创新性。

本文以自顶向下的视角，在交互手势集与识别算法方面进行创新，构建了一套智能手表周边手势识别的系统，拓展了用户与智能手表的交互方式，基于创新、实用的二元悖论问题提出了一种解决方案，相比过去的相似研究更为系统。

本文工作内容如下。

（1）智能手表周边手势引导研究（Gestures Elicitation Study, GES）。针对目前对于适配运动传感器进行识别的智能手表周边手势集空白问题，本实验进行了手势引导研究，收集了实验参与者设计的智能手表周边手势。此类手势使用户在智能手表上的操作不再拘泥于传统的屏幕触碰。本实验利用李克特量表统计方法以及同意值等指标，对收集的手势按照指令视角进行统计分析。

（2）利用加速度计数据进行用户静止状态下的智能手表周边手势识别。目前的主流实验中，用于识别手势的传感器多为非常见的类型。本实验基于几乎所有智能设备都具备的加速度计，采集了静止状态下实验参与者做出手势时，三轴加速度计产生的值变化，并利用循环卷积神经网络（Convolutional Recurrent Neural Network，CRNN）对于时间序列数据的处理优势，完成多分类任务，在保证平均识别准确率接近 90%的前提下，做到传感器成本更低，模型更易部署。

（3）利用智能手表进行用户运动状态下的手势识别。这一部分的实验将应用场景拓展到用户身体运动状态下的手势识别，并且增加了数据来源，将智能手表中常见的陀螺仪也纳入数据处理的范围，进一步增加模型的可靠性和实用性。7 种手势中最高的识别率达到 77%，在运动状态下微弱手势的识别这个空白的领域，初步具备了应用的可能性。

关键词：智能手表，手势引导，手势识别，人机交互

Abstract

Wearable devices have brought people richer and more convenient experiences in their daily lives. While new devices offer more multimedia functionalities, the wearable experience for users encompasses not only these new multimedia features but also factors such as operational comfort and convenience, among other Human-Computer Interaction (HCI) elements. Traditional smart devices interact with users through screens, but the advancement of sensor technology has made gesture-based interactions around smart wearable devices gradually possible. While scholars have delved into this area in the past, they often struggle to balance the practicality of the technology and the innovativeness of the gesture set, hindered by several factors such as algorithm limitations and sensor accuracy.

This paper adopts a top-down perspective and innovates in the aspects of interaction gesture set and recognition algorithm, constructing a system for around-smartwatch gestures recognition. It expands the interaction methods between users and smartwatches and proposes a solution to the binary dilemma of innovation and practicality, which is more systematic compared to similar past research.

The work of this paper is outlined as follows:

(1) Gestures Elicitation Study (GES) for smartwatch peripheral gestures: Addressing the current blank space in gesture sets for smartwatch peripheral gestures suitable for recognition by motion sensors, this experiment conducted a gestures elicitation study, collecting gestures designed by participants. These gestures liberate users from traditional screen touches when operating smartwatches. The experiment utilized Likert scale statistical methods and agreement metrics to statistically analyze the collected gestures from an instructional perspective.

(2) Recognition of around-smartwatch gestures under user static states using accelerometer data: In mainstream experiments, sensors used to recognize gestures are often of uncommon types. This experiment, based on the accelerometer present in

almost all smart devices, collected changes in values generated by the three-axis accelerometer when participants made gestures under static states. Leveraging the advantages of Convolutional Recurrent Neural Networks (CRNN) in processing time series data, the experiment completed multi-class classification tasks. While ensuring an average recognition accuracy close to 90%, it achieved lower sensor costs and easier model deployment.

(3) Recognition of user gestures under motion states using smartwatch: This part of the experiment extended the application scenario to recognize gestures under user body motion states and included additional data sources. It incorporated the common gyroscope found in smartwatches into the data processing scope, further enhancing the reliability and practicality of the model. The highest recognition rate among the seven gestures reached 77%, showing preliminary potential for application in the blank area of recognizing faint gestures under motion states.

Keywords: smartwatches; gestures elicitation; gesture recognition; Human-computer interaction

Contents

List of Tables

List of Figures

Chapter 1. Introduction

Smart devices are revolutionizing our lives by substantially improving production efficiency and facilitating various daily activities. Smartwatches have risen in popularity during the 2010s. According to studies from Statista revenue in smartwatches are estimated to reach 47.95 billion dollars in 2024 and revenue per year is expected to continue to grow to 62.46 billion by 2028 [1]. Current market share is a three-way split between Apple, four second-tier brands like Huawei and Samsung, and about two-fifths of the market split among dozens of smaller vendors [2].

Smartwatches are widely used in the field of communication because of their small and convenient advantages and the function of transmitting information, the most common is message viewing, voice assistant and so on. In addition, with the development of sensor technology, more and more body detection-related sensors are incorporated into smart watches, making them more and more important in sports health, medical health [3], and other aspects, with the current technology, smartwatches can not only complete the step detection and other functions, and even detect cardiovascular health [4].

According to some surveys, the research field of Human-Computer Interaction (HCI) develops designs, conducts user studies, and evaluates interactive computing systems, focusing on improving usability and user experience [5, 6], like speech recognition [7]. Smartwatches are important devices in HCI usage scenario. Here are two major constraints of smartwatches compared to smart phones. To keep them small enough to wear on a wrist, they usually have small screen, which results in firstly restricted I/O methods, secondly weaker computing capability and limited battery capacity. While smartwatches have two strong advantages: mount location and continual connection to the skin. Former one means that sometimes they do not need to occupy both hands, and latter one helps activity recognition by removing the burden of identifying the location of devices [8].

1.1 Interaction methods

HCI technologies have played a remarkable role in the smartwatches use. Quick and simple gesture operations are becoming increasingly popular among smart device users. Researchers are exploring ways to enhance the user experience through these gestures. Here are two main interaction methods applied on smartwatches.

1.1.1 Traditional interaction

Users interact with smartphones by touching numerous virtual buttons on the screen; they can even input text via a virtual keyboard. Furthermore, they can zoom in and out to focus on the entirety or details. A large screen area enables these inter-actions.

In the early days of smartwatch development, it was widely seen as a shrunken version of a smartphone, but with some complex features cut off. Therefore, the operation of clicking and sliding fingers on smartphones is also naturally applied to the interaction of smartwatches.

Take the OPPO Watch 2 (42mm with Bluetooth) as an example, it has two physical buttons on the right side, one is used to light the screen, can also be used to enter the application list interface, and the other button is responsible for bringing up the background program list. The rest of the action is done on the watch face, which users can tap or swipe with their fingers as they would on a smartphone.

There is also a device on the market called the Smart band, which has a smaller screen than the average smartwatch, is more streamlined, and cannot run complex programs like which you can run on a smartwatch. Take the Huawei Band 2 as an example, it only sets a button on the right side to control the screen switch. Other operations are conducted on the screen. Due to the small screen, the click operation can only be partially positioned, while the sliding operation cannot be accurate to the specific position. On the whole screen, the up and down slide can control the change of the list content, and the left slide represents the back or exit.

Figure 1.1 shows a smart band and a smartwatch.

Figure 1.1: Smart devices

This operation leads to the dilemma of smart wearable device being used as auxiliary tools. Most smartwatches can only be used as simple controllers attached to smartphones, and even some smart wristbands can only be used as human data collectors. Of course, this is another function of smart wearable devices independent from smartphones.

The technology studied in this thesis can be applied to both smartwatches and smart bands, so smart wearable devices are no longer distinguished in the following text, and they are collectively referred to smartwatches.

1.1.2 Advanced interaction

Major manufacturers continuously release new models, driving rapid development in the smartwatch industry. However, the small screen-size of a smartwatch limits what users can do on them compared to smartphones. The most common gestures used to operate smartwatches is touching the screen. However, this gesture has some limitations, including a limited number of virtual buttons due to the small operation area and difficulties in usage when fingers cover the screen. Researchers are studying single-handed operations [9, 10] and around-smartwatch gestures [11, 12] to overcome these issues and improve user experience.

(1) One-handed gestures. Also known as single-handed gestures. All the operation would be done by the hand wearing the smartwatch. Typical gestures including rotating the wrist, waving the arm and bend the fingers. Usually combined with head-mounted displays in virtual reality region [13].

(2) Around-smartwatch gestures. This kind of gesture requires users to employ two hands, one for wearing the smartwatch while the other to interact with the neighbor area of the smartwatch. Typical gestures including making a hand posture around the smartwatch and getting closer to the smartwatch from a specific direction.

1.2 Sensor types

When the user gestures, it is usually necessary to use some technology to recognize these gestures. In recent years, scholars have tried many different methods. This thesis mainly introduces several mainstream technologies here, including image processing, electric field sensors and motion sensors.

1.2.1 Image

Usually use some image tools like camera. With the development of image processing technology, this technique is now being widely used. When a user performs a gesture, an external camera captures the moment of the gesture and then uses image recognition technology to classify different gestures. The advantage of this technology is that the accuracy of gesture classification is high, and it can adapt to a variety of gestures. However, the disadvantages are also obvious, the equipment deployment cost is high, the need for additional external equipment to assist, in addition, the deployment ability of large neural network model is poor, it is difficult to transplant the model to mobile devices characterized by portability.

Yan et al. [14] employed three cameras to recording the gesture video. They set twenty-one hand-landmarks on a hand; three cameras would record the videos from three different directions. 252 features were created from the coordinates of landmarks and other objects. K-means++ algorithm was used to cluster the gestures. This example shows a usage of image.

VGG-16 is a deep convolutional neural network architecture, which is composed of sixteen convolutional layers and fully connected layers and is one of the VGGNet series. VGG-16 has achieved reliable performance in ImageNet image classification and has become one of the benchmark models for many computer vision tasks. Ding et al. used VGG to process gesture data in the form of image data, and the recognition rate is higher than other models [15].

Image recognition is a commonly used gesture recognition method, but it is often accompanied by prohibitive cost, so we give up using this method for research.

1.2.2 Electric field

The electric field is everywhere, and the activity of the finger around the device will drive the change of the electric field in the environment [16]. The electric field sensor records the value through such changes and fluctuations, and then realizes the purpose of gesture recognition through some models. Gesture classification by electric field sensors has the advantage of high accuracy, but cost is still a pain point, the scope of application of this sensor is narrow, and almost all smart watches on the market are not equipped with this sensor, which makes the realization of this technology has become far out of reach.

1.2.3 Motion sensor

Motion sensors are one of the most common types of sensors in daily life, and smart devices are usually equipped with at least one accelerometer and a gyroscope, both of which are motion sensors. Smart devices use this kind of sensors to complete the number of steps taken by users each day. Motion or gesture capture with these sensors is easy and inexpensive.

For example, Kamachi et al. [17] used accelerometer based on a smartwatch to predict eating activity, interestingly, they succeed to recognize the food category by SVM, even the accuracy is not high enough for application.

Due to the lack of public data set based on inertial measurement unit, some scholars have tried to convert gesture video data into accelerometer and gyroscope data through video public data set conversion. The feasibility of this transformation from the side confirms the convenience of using motion sensor data for recognition tasks [18].

However, the disadvantage is that when the change in the posture of the device caused by gestures is small, the numerical change of the sensor is not obvious, and compared with other sensors, it is more susceptible to noise pollution, so it is difficult to identify subtle gestures.

1.3 Overall structure

The main challenge is finding more users preferred around-smartwatch gestures and a cost-effective method to identify around-smartwatch gestures. Some researchers have proposed gesture sets without including the recognition solution, while some have used expensive sensors like electromagnetic field sensors.

Here is much work needed to be done as we want to build a new human-smartwatch interaction system. First, as the around-smartwatch interaction being a new field in human-computer interaction, we should investigate in user preferences. Given a simulated application scenario and typical command, users would design some gestures suitable for corresponding commands, we aim to collect users' preferences, then consider the most reasonable gestures set.

Figure 1.2 shows the overall structure.

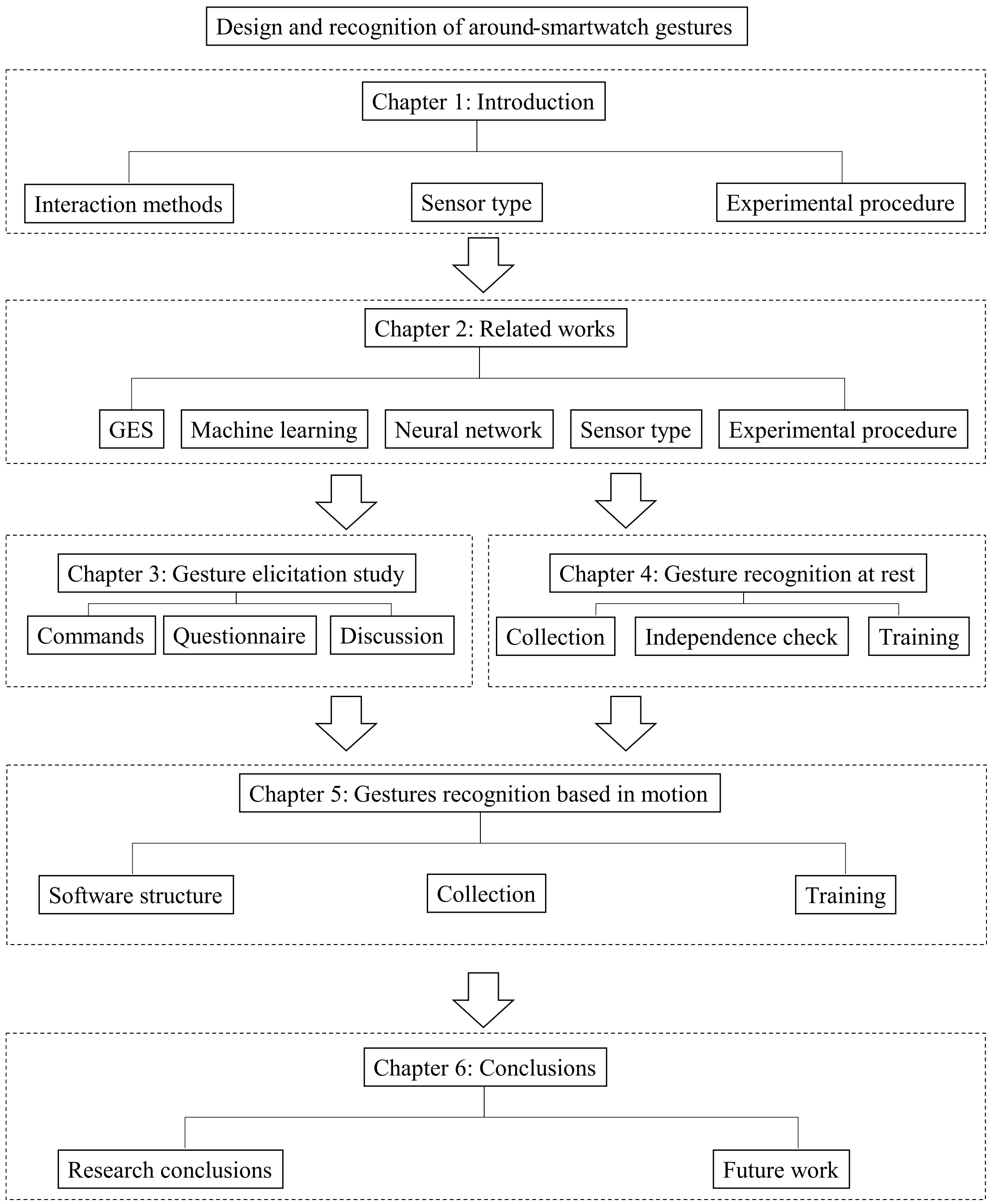


Figure 1.2: Overall structure

Beside this, another critical issue is the way to recognize gestures. We have addressed this issue by exploring gestures using affordable sensors. To validate the possibility of recognizing around-smartwatch gestures based on an accelerometer, we tried to collect the signal by Arduino accelerometer firstly and designed a neural network as recognizer to recognize the gestures. Participants wore a set of equipment that simulated a smartwatch, and signals from the accelerometer were recorded and labeled. Then, recurrent convolution neural networks were trained to recognize these gestures effectively.

After getting the satisfactory results of recognition based on an accelerometer, we tried to transfer the data collection experiment on a real smartwatch, which including not only accelerometer, but also gyroscope. The dimension number increases, which can contribute to higher accuracy and more gesture candidates. We code sensor listener program for a real smartwatch as well as data transmission and storage program on a smartphone. Just like what we did in previous step, we also trained networks as recognizer.

After finishing all the works, we made the conclusion of our study.

Chapter 2. Related works

2.1 Gestures elicitation studies

Elicitation, according to a review in 2020, is the process of making participants respond to referents and propose gestures to affect those referents [19]. Gestures elicitation study (GES) is typically a method for researching the use and generation of gestures. This research often involves observing and analyzing the gestures people use in specific contexts to understand their meanings, purpose, and cultural backgrounds. Such studies help reveal the role and impact of gestures in communication.

GES has many implications. For users, it can enlighten them about new methods of human-computer interaction and expand gesture cognition. For application developers, it helps them explore new user needs. For equipment manufacturers, technology can be updated in a timely manner.

In 2005[20], Wobbrock et al. proposed “guessability method”, which is a metric designed to measure users’ preferences of symbolic input. Kim et al. [21] used this method to design gestures set in deformable displays interaction. A metric “Agreement score” was proposed to indicate the most preferred gestures performed by participants.

Kerber [22] et al. tried to elicit users preferred gestures on a certain context, they deployed a smartwatch as a base recognizer, set up a simulating scenario like listening to music while walking on a street, asked participants to invent some single-handed gestures based on an arm or fingers. They called their research as same-side interaction. Agreement score was also calculated to measure the results of users’ preferences.

To elicit motion gestures, Ruiz et al. [23] presented nineteen tasks including actions on call, like answering a call or ignore a call, and navigation commands, like choosing vertical next item or zooming in. They asked participants to focus on preferences excluding the influence of recognizer technology. After that, they calculated the agreement score to determine the end-users' defined gestures.

Wobbrock's work has enlightened many gestures elicitation studies. One of the well-known indicators is Agreement score. Like previous studies, our study also used this indicator to explore users' preferred gestures. The difference is that our research focuses on a new field, around-smartwatch gestures, and developed a new recognition model based on neural networks to make it possible to recognize around-smartwatch gestures.

In the field of gesture elicitation studies, many scholars have similar research methods, but there are many branches of gesture type subdivision. Some scholars use smartwatches to detect human activities, such as running, ball sports, etc. To a smaller extent, virtual reality technology is inseparable from user gesture interaction, reasonable gesture design is crucial. To the more microcosmic, it is the use of wrist and finger changes to design gestures, which provides innovation in the application layer.

The gestures around smartwatches studied in this thesis focus on areas that are more difficult to be recognized, extending the channel of human-computer interaction to the skin area around smartwatches.

2.2 Machine learning

Machine Learning (ML) is a discipline dedicated to the development of computer systems capable of learning from experience and continually improving their performance. The fundamental objective is to enable computer systems to extract patterns and regularities from data and utilize this knowledge to make predictions or decisions without explicit programming.

Machine learning algorithms can be broadly categorized into two main types: Supervised Learning, and Unsupervised Learning. In Supervised Learning, algorithms are trained with input-output pairs to establish a mapping between inputs and outputs, enabling accurate predictions on new data. Unsupervised Learning aims to discover hidden structures or patterns within unlabeled data, such as clustering or dimensionality reduction.

Within these major categories, machine learning encompasses various classic algorithms, including Linear Regression, Support Vector Machines, Decision Trees, Neural Networks, Clustering Algorithms, among others. These algorithms exhibit distinct advantages in different tasks and scenarios, finding widespread applications in image recognition, natural language processing, medical diagnosis, financial forecasting, and other domains.

2.2.1 Decision tree

In the field of machine learning, decision tree algorithm was first proposed by Ross Quinlan. In his 1986 article, Quinlan introduced “Iterative Dichotomiser 3” algorithm. The algorithm classifies samples according to data characteristics and constructs a decision tree by recursively selecting the best features for data partitioning. His research has played a significant role in the development of decision tree algorithms.

Figure 2.1 shows a procedure to judge the quality of a watermelon using the decision tree algorithm.

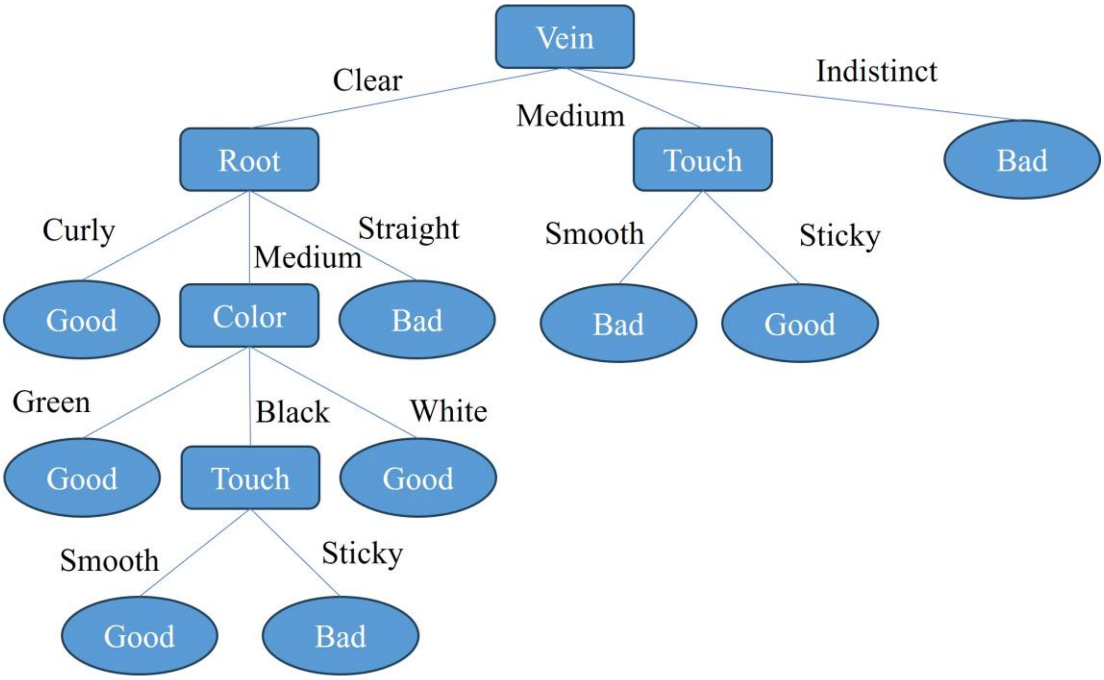


Figure 2.1: Decision tree

2.2.2 Ensemble learning

Ensemble learning is a machine learning approach that aims to improve overall performance and generalization by combining predictions from multiple basic models. The core idea of ensemble learning is that a combination of multiple weak learners may be more expressive and robust than a single strong learner. Ensemble learning is composed of several basic learners, the common basic learners include decision tree, support vector machine, neural network and so on. His results can be obtained by averaging or voting the predictions of multiple models.

Random forest is an ensemble learning method that improves the performance of an overall model by combining the predictions of multiple decision trees. Its major features include using random sampling to generate multiple different decision trees, and randomly selecting a part of the features on each node for partitioning.

Tin proposed “random decision forests” algorithm in 1995 [24], which is the first algorithm of random forests, he used random subspace method. In 2001, Leo developed extensive algorithm for random forests [25], Adele registered “random forests” as a trademark in 2006.

Due to its efficient and powerful performance, random forests have become very popular in practical applications, especially for classification and regression problems.

2.3 Neural network

The concept of neural networks can be traced back to the 1940s and 1950s. Warren McCulloch and Walter Pitts proposed a theoretical model called an "artificial neuron" in 1943, attempting to simulate the functioning of biological neurons.

Building on this, Frank Rosenblatt introduced the perceptron model in 1958 [26], a simple neural network structure used for binary classification tasks. However, due to the limitations of perceptron, progress in neural network research during that time was not as rapid as anticipated.

The real impetus for the development of neural networks came in the 1980s and 1990s with the introduction of the backpropagation algorithm and other improvements, leading to wider applications and research in neural networks.

With advancements in computational capabilities, the availability of big data, and improvements in deep learning algorithms, neural networks have experienced quick progress in recent years. They have achieved remarkable success across various fields.

2.3.1 Convolutional neural network

Based on neural network, LeCun proposed Convolutional Neural Network (CNN) [27], which is a kind of deep learning model used to process and analyze data with network structure, the most typical application is image recognition.

The convolution layer is the core component of the CNN, which extracts local features by applying a learnable filter or convolution kernel to a local region of the input data, where the activation function can introduce nonlinear features and increase the ability of expression of the model. The pooling layer is used to reduce the dimension of the feature map, thus reducing the number of parameters and the amount of computation. Common pooling operations include maximum pooling and average pooling. After the multi-set convolution layer and pooling layer, CNN usually use the fully connected layer to classify or regression the extracted features. The training process of CNN usually adopts the backpropagation algorithm to update the network parameters by minimizing the loss function, so that the model can gradually learn the feature representation of the input data and task-related knowledge. Stochastic gradient descent or its variants are generally used to optimize parameters during training.

CNN has achieved remarkable success in the field of computer vision, widely used in image classification, object detection, image segmentation and other tasks, and has achieved excellent performance in many competitions and practical applications. At the same time, CNN has also been applied in other fields, such as natural language processing, speech recognition, etc., showing its strong generalization ability and application potential. Figure 2.2 shows a structure of 2D convolutional network.

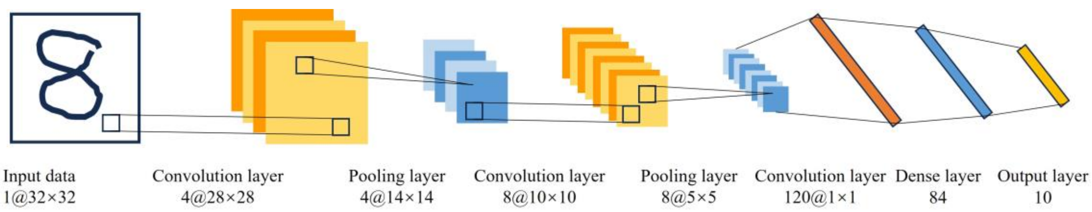


Figure 2.2: Convolutional neural network

2.3.2 Recurrent neural network

Recurrent Neural Network (RNN) are a type of neural network architecture that is highly effective in handling sequential data. Compared to traditional feedforward neural networks, RNNs utilize their internal recurrent structure to maintain memory of past information in the sequence, thereby better adapting to the characteristics of sequential data.

Ising proposed Ising model in 1925, which is the first RNN architecture that did not learn. Shun'ichi Amari made it adaptive in 1972 [28], this network was also called Hopfield network [29]. Figure 2.3 shows the structure of RNN.

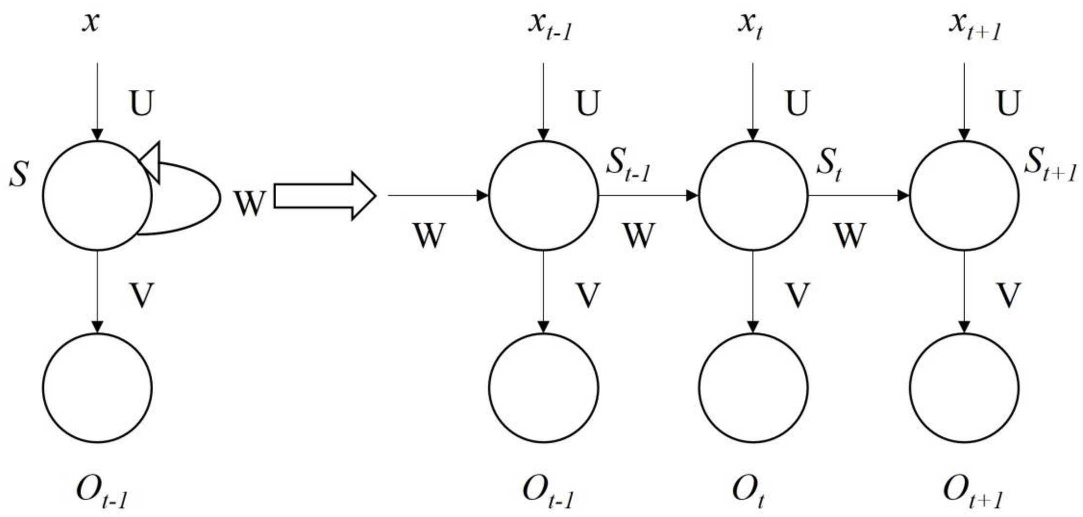


Figure 2.3: Recurrent neural network

The core idea of RNNs is to propagate information from one time step of the sequence to the next while retaining processing of historical information. In an RNN, each time step's input includes not only the current input data x^t but also the hidden state from the previous time step h^{t-1}. This propagation of hidden states allows RNNs to have memory capabilities, capturing long-term dependencies in the sequence.

Specifically, the computation process of an RNN is as follows:

At each time step t, input data x^t and the previous time step's hidden state h^{t-1} are input into the RNN.

The RNN uses the input data x^t and the previous time step's hidden state h^{t-1} to compute the current time step's hidden state h^t, usually applying an activation function (such as tanh or ReLU) to process the result.

The current time step's hidden state h^t can be used for prediction at the current time step, propagated to the next time step, or used for other tasks.

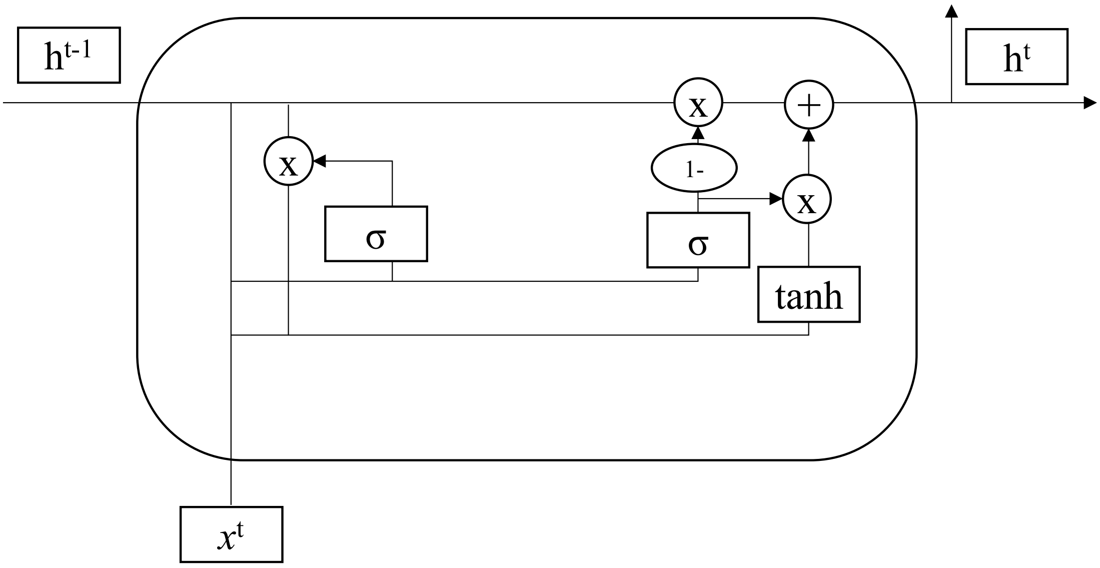


Figure 2.4: Gated recurrent unit

However, traditional RNNs suffer from the vanishing or exploding gradient problem when dealing with long sequences, making it difficult to capture long-term dependencies. To address this issue, several improved RNN structures have been developed, such as Long Short-Term Memory (LSTM) [30] and Gated Recurrent Unit (GRU). Figure 2.4 shows the structure of GRU.

LSTM and GRU introduce gate mechanisms to control the flow of information, effectively addressing the gradient issues of traditional RNNs. These gate mechanisms selectively allow information to pass through while suppressing unnecessary information flow, enabling the network to better adapt to the characteristics of long sequence data.

In summary, Recurrent Neural Networks are a powerful tool suitable for processing several types of sequential data, including natural language processing, time series prediction, audio processing, among others.

2.4 Around-smartwatch gestures

Extensive research has been conducted on smartwatch gestures, particularly single-handed gestures. For instance, Yu et al. [9] developed a method using electromyography (EMG) sensors and a RNN to detect gestures like circling using fingertips, while Wen et al. [10] employed an accelerometer and a sup-port-vector machine (SVM) to identify finger gestures like pinching, tapping, and rubbing.

Kim et al. [31] combined the single-handed around-smartwatch gestures with the vehicle control system, set up a total of eight gestures, collected data from the accelerometer and gyroscope of the smartwatch worn by the participants, and then trained the model with a neural network. They proposed an UI system that uses the lap as an interaction surface to enable 2D touch interaction in a vehicle environment.

To overcome challenges like finger occlusion, researchers have proposed around-device interaction. Zhou et al. [11] implemented virtual buttons around a smartwatch using an electric field sensor and SVM, achieving an accuracy of ~90%. Another approach [12] involved using photo reflective sensors for gestures like pushing and pinching as well as SVM. Kwon et al. [37] user the accelerometer in a smartwatch for gesture recognition of the nine-square line.

As this study focuses on an affordable accelerometer, not all gestures from previous research are applicable. The authors specifically chose relevant gestures for accelerometer recognition.

2.5 Time series data processing

DTW (Dynamic Time Warping) is a method for comparing the similarity between two time series. It considers the timing information between sequences and allows the sequence to be extended or contracted on the timeline to find the best match. DTW is often used in speech recognition, handwriting recognition, bioinformatics, and other fields, and can effectively cope with the different length of time series or the existence of a certain degree of time deviation [38].

An IMU usually includes an accelerometer and a gyroscope. Liu et al. [39] employed IMU in both smartwatch and smart ring to detect finger motion. They applied low pass filter to the raw data to preprocess.

Chen et al. [40] used deep learning to identify human activities like running, jumping, and walking by creating an Android application that uses the smartphone accelerometer. Their three-layer CNN outperformed the SVM and deep belief network, achieving an accuracy of 93.8%. The movements involved significant amplitude motions, contributing to the high accuracy due to their distinct signal ranges.

Koch et al. [41] employed accelerometers around the forearm instead of costly sur-face electromyography (sEMG) systems. Their RNN, which featured LSTM and convolution LSTM cells, considered individual physiological characteristics, resulting in increased accuracy with a 5-ms window length.

Bloch et al. [42] used CNN and transfer-learning based classification model to handle accelerometer signals. They applied some pre-processing method like filters, normalization, and augmentation on the raw data, they compared the performance of some deep CNN models, and used F1 score to measure the performance. Their innovation was used for feeding behavior classification of dairy cows.

Khanh Nguyen-Trong et al. [43] designed a set of hand gestures called Geshome, like moving left, moving down, and drawing a number. They proposed two deep convolutional LSTM architectures, CNN-LSTM and CNN-biLSTM, all convolutional layers were 1D-CNN instead of 2D-CNN. They evaluated several datasets and several network, 1D-CNN-biLSTM got the highest F1-score in all tasks.

Recognizing gestures around a smartwatch is challenging owing to the trade-off between low hardware costs and high accuracy. This study aims to improve recognition accuracy while simultaneously ensuring affordability.

2.6 Contribution

One contribution is that we expand the interaction gestures of smartwatch devices. This thesis focuses on the gesture’s elicitation around a smartwatch. In the past, few scholars have done research in this area, possibly because inefficient gesture recognition systems make these new gestures impractical. However, our recognition system can solve this problem. In addition, their gesture set is more of a specialized gesture set for other sensors, such as electric field sensors that can detect an object's approach or departure. However, the gesture set in this thesis makes the recognition ability of accelerometer and gyroscope special, so it is more practical.

Another contribution of this study is the introduction of a new gesture recognizer that uses ensemble convolutional recurrent neural networks. This technology recognizes around-smartwatch gestures in a cost-effective manner. This approach involves using a type of neural network that can recognize gestures using the built-in accelerometer and gyroscope. With most smartwatches equipped with accelerometers and gyroscopes, this recognition technology can be immediately implemented for practical applications.

Chapter 3. Gestures elicitation

3.1 Scenario and command

Our proposed around-smartwatch gestures are operated on the skin next to the smartwatch and have a smaller range of physical activity than gestures in other human-computer interactions. This characteristic determines that such gestures are more suitable for human-computer interaction scenes with high frequency in daily life. Compared with applications such as VR games, such gestures are more suitable for human-machine control in daily life, such as mobile phone calls, headset playback and other scenes. Therefore, we propose a simulation scenario, which is the control of the music player. The figure 3.1 shows a series of commands we set in the music player application scenario, including backward, previous song, play, pause, next song, fast forward, lower volume, increase volume, a total of eight commands.

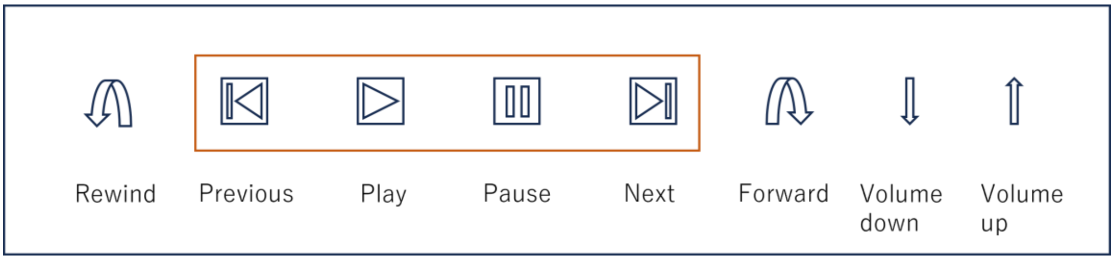


Figure 3.1: Commands set

The application scenario is assumed to be the state of running. While running, users wear a pair of Earphones to listen to music. Earphones, smartwatch, and smartphone are connected to each other through Bluetooth. When the user wants to operate the music player, such as turning down the volume, cutting songs, and so on, since the mobile phone is in the pocket, it is inconvenient to take out, and the interactive button on the smart watch can only realize limited functions, while the volume cannot be adjusted, so the user can complete the interaction with the smartwatch through the around-smartwatch gestures, so as to achieve the purpose of operating the music player. In this case, the user only needs to change from running to brisk walking, and then raise the wrist to make gestures.

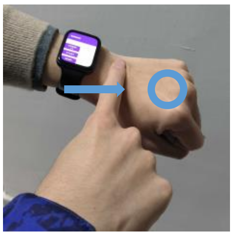
3.2 Instruction and questionnaire

Participants were asked to simulate a brisk walk and wear a smartwatch on their left hand, with their wrists flat on their chest and trying their best to maintain a steady posture. For each instruction, the participant needs to design a gesture that he thought was appropriate to corresponding command and write down the description of such gesture in the questionnaire. Meanwhile, we also recorded videos of the participants performing the gesture.

In the questionnaire, for each command, we asked the participants to fill in the gesture they designed and collect the confidence scores of the participants about each gesture they designed. In the confidence section, we used a 7-points Likert Score. The statement is that you think the gesture you designed is suitable for performing the corresponding command control. If the participant thinks it is very suitable, then choose "Strongly agree", if the participant thinks it is very unsuitable, but cannot produce a more appropriate gesture, then choose "Strongly disagree".

3.3 Result

We invited 14 participants, they are all male, between the ages of 22 and 25. Each participant is presented with the effect of each function, e.g., stopping the current music to play the next music in playlist for “next” button, and the participant is asked to propose a gesture around smartwatch that would generate the effect just witnessed. At the end of the study, we collected a set of 112 gestures = 8 (commands) × 14 (proposals). We looked at the set of gestures elicited for each command to understand whether there are any gestures in agreement. If there is significant agreement among participants, and they represent the target population of smart ring users, we can be confident that the gestures are intuitive. Inexperienced users, who were not part of the original study, are likely to easily learn and possibly prefer the same gestures.


(a) Sliding and hold


(b) Sliding downwards


(c) Sliding upwards


(d) Sliding towards left


(e) Sliding towards right


(f) Integrated sliding


(g) Draw a circle


(h) Tapping once


(i) Tapping twice


(j) Tapping for three times


(k) Pressing and hold


(l) Stop running and walking


(m) Putting down the arm


(n) Rotating the wrist inward


(o) Rotating the wrist outward


(p) Waving the arm upwards


(q) Waving the arm downwards


(r) Waving arm with a specific posture

Figure 3.2: Collected gestures

We designed a table to record the designed gestures and confidence of each participant. Since the gestures designed by each participant were different, we classified the gestures collected into five categories: finger sliding, finger tapping, wrist rotation, arm waving, and posture adjustment. Each class has subclasses, such as finger sliding which includes sliding upwards, sliding downwards, sliding towards left, sliding towards right, integrated sliding, and drawing a circle.

Some participants designed similar but not the same gestures. We classified these gestures according to their key features. Similar gestures were not treated as a separate category of gestures but were grouped into the same subclass. After reviewing all the gestures from all the participants, we grouped these gestures into eighteen subclasses. Figure 3.2 shows all the gestures designed by participants.

In addition, we designed a table of commands views, which count the preferences of various gestures in each command.

$$A = \sum_{r \in R} \sum_{P_i \subseteq P_r} \left(\frac{|P_i|}{|P_r|} \right)^2 \cdot 100\% \tag{3.1}$$

Agreement score is introduced by Wobbrock in 2005 [44], it is a metric that can be used to measure the agreement degree of participants' proposed gestures type upon a command. Intuitively, agreement score should be 100% when proposed gestures are identical and be nearly 0% when they are unique. In this equation, A represents agreement score, P represents captured set, R represents all referents. For example, in fourteen proposals for command volume down, 9/14 are sliding by a finger, 4/14 are rotating by wrist, 1/14 is waving the arm, then agreement score should be 50%. Table 3.1 shows the result of the questionnaire.

Table 3.1: Result

	Votes	Average confidence	Agreement score
Rewind			0.23
Sliding and hold	1	6.00	
Sliding upwards	1	6.00	

Sliding towards left	6	6.00	
Drawing a circle	1	5.00	
Tapping once	1	4.00	
Pressing and hold	1	6.00	
Rotating the wrist inward	2	5.50	
Waving arm with a specific posture	1	6.00	
Previous			0.17
Sliding upwards	1	7.00	
Sliding towards left	4	6.25	
Sliding towards right	1	6.00	
Integrated sliding	1	6.00	
Tapping for three times or more	3	5.33	
Pressing and hold	1	7.00	
Rotating the wrist inwards	2	5.50	
Rotating the wrist outwards	1	7.00	
Play			0.39
Tapping once	8	5.88	
Tapping twice	3	6.33	
Pressing and hold	1	5.00	
Rotating the wrist outwards	1	6.00	
Waving the arm upwards	1	3.00	
Pause			0.23
Sliding downwards	1	6.00	
Tapping once	5	5.60	
Tapping twice	4	6.75	
Pressing and hold	1	5.00	
Stop running and walking	1	4.00	
Rotating the wrist outwards	1	6.00	
Waving the arm downwards	1	6.00	

Next			0.15
Sliding downwards	1	6.00	
Sliding towards left	1	7.00	
Sliding towards right	4	6.00	
Integrated sliding	1	6.00	
Tapping twice	2	6.50	
Tapping for three times or more	1	6.00	
Pressing and hold	1	6.00	
Rotating the wrist inwards	1	7.00	
Rotating the wrist outwards	2	5.00	
Forward			0.29
Sliding and hold	1	6.00	
Sliding upwards	1	6.00	
Sliding towards right	7	5.71	
Drawing a circle	1	5.00	
Tapping once	1	4.00	
Pressing and hold	1	5.00	
Rotating the wrist outwards	1	7.00	
Waving arm with a specific posture	1	6.00	
Volume down			0.50
Sliding downwards	9	6.67	
Rotating the wrist inwards	4	6.00	
Waving the arm downwards	1	6.00	
Volume up			0.50
Sliding upwards	9	6.67	
Rotating the wrist outwards	4	6.00	
Waving the arm upwards	1	6.00	

After collating the questionnaire data, we calculated the agreement score of the eight commands, as well as the mean and standard deviation of trust. These data are presented in the table.

3.4 Discussion

3.4.1 Gesture assignment

According to the table 3.1, the commands in order of Agreement score are Volume down, Volume up, Play, Forward, Rewind, Pause, Previous, and Next. For each command, we would use the gesture which gains the most votes. But here may be a problem, which is that the mostly voted gesture in different commands may be the same one. For example, among rewind command and previous command, the gesture with the most votes is Sliding towards left. Therefore, according to the order of the Agreement score, The Rewind command is first assigned the Sliding towards left gesture, while the Previous command assigned the Tapping for three times, which has the secondary votes. One exception is, there is no conflict between Play and Pause, so they are assigned the same gesture. All gestures are assigned as shown in figure 3.3.

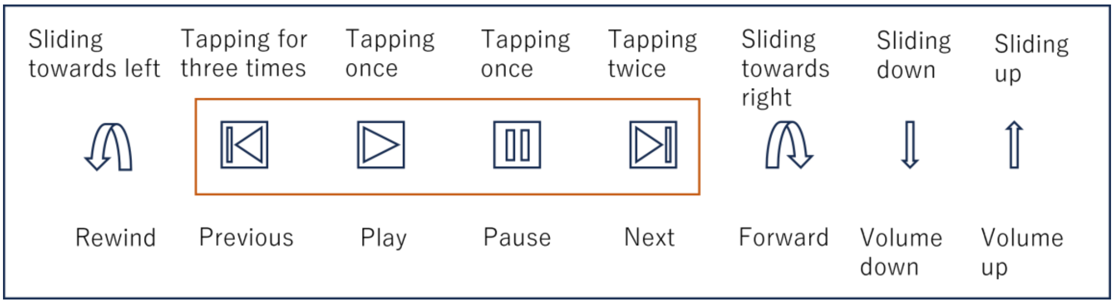


Figure 3.3: Gestures assignment

Based on each command, we compared the confidence of each gesture. For example, in the Rewind command, a variety of gestures have gained six points of confidence, but only one gesture, sliding towards left, not only gained the highest confidence score, but also gained 6 votes. From this, we can see that for the Rewind command, users are more inclined to use sliding towards left, and is confident in the representativeness of the gesture. A similar example is the sliding upwards gesture in the volume up command. Each gesture's vote count and confidence factor into how we determine the relevance of the commands to the gestures.

The mean and standard deviation of confidence score are also calculated for each command. Given that a participant may design several different gestures in response to a given command, confidence score is not a measure of how good a gesture is, but rather how the participant treats a given command. For example, the fact that Volume down and Volume up have the highest average score indicates that participants are good at designing gestures corresponding to these two commands. In other words, these two commands are intuitive from the user's perspective and close to the application scenario. The standard deviation measures the difference in participants' interaction habits with a given instruction. For example, the standard deviation of the Play instruction is the highest among all the commands, which can indicate that some users have a very habitual operation mode for this command, and some users think that there are diversified gesture design ideas for this command.

3.4.2 Confidence

If the agreement score represents the degree of consensus among experimental participants regarding the design of new interaction gestures for a certain command, then the confidence score represents the degree of agreement of the experimental participants with a particular gesture. From the perspective of commands, a higher confidence score indicates that the corresponding gesture for that command is easier to design, whereas a lower score suggests difficulty in design. From the perspective of gestures, a high confidence score for a particular gesture indicates that users are familiar with it and find it easier to operate. The information conveyed by such gestures is highly consistent among multiple users. This aligns with one of the purposes of our research—to design gestures that are preferred by users for their ease of execution and understanding.

Chapter 4. Gesture recognition at rest

4.1 Gestures set

There are many public datasets about human activities or gestures, such as the Opportunity Dataset collected from residents' morning activities [45], the Skoda Dataset collected from assembly lines, and the UWaveGesture Dataset collected from remote control scenarios [46]. And the HCI Dataset collected from human-computer interaction scenarios [47].

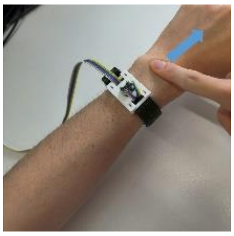
However, those public datasets are not suitable for the research in this thesis, because the application scenarios are different. The gestures elicited in previous chapter appear less movement, those models who perform well in public datasets may perform worse in the scenario in this thesis, thus we decided to abandon those public datasets.

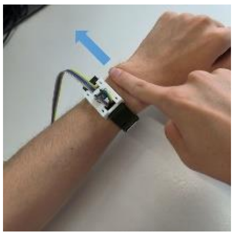
Based on the experiments in the previous chapter, we set up a gesture set for gesture recognition experiments. It contains all gestures except for three times tapping but include two gestures from the drawing a circle class that participants had suggested but not accepted. Preliminary experiments will explore the validity of these gestures. In addition, we also set up a static gesture to serve as a control group. The preliminary experiments were conducted with participants at still, not in a way that mimics brisk walking. The success of the gesture recognition experiment in this step will pave the way for further scenario expansion. Figure 4.1 shows the gesture appearances.

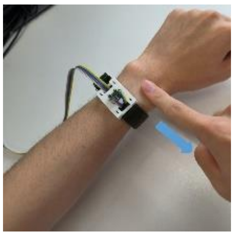
Table 4.1: Collected gestures

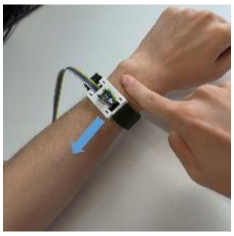
No.	Name	Description
0	keeping still	keep still
1	sliding right	slide the skin towards right at the right side of a watch

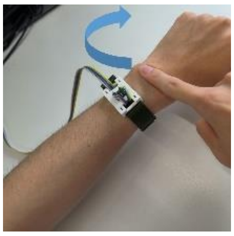
2	sliding up	slide the skin towards up at the right side of a watch
3	sliding down	slide the skin towards down at the right side of a watch
4	sliding left	slide the skin towards left at the right side of a watch
5	drawing a circle	draw a clockwise circle on the right side of a watch
6	drawing an anticlockwise circle	draw an anticlockwise circle on the right side of a watch
7	taping	tap on the skin of the right side of a watch
8	double tapping	double tap on the skin of the right side of a watch

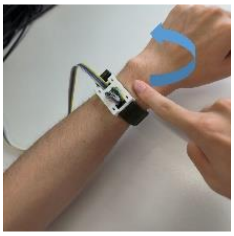

(a) sliding right

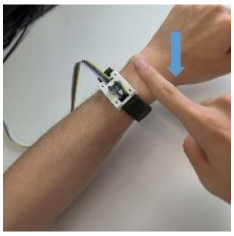

(b) sliding up


(c)sliding down


(d) sliding left


(e) drawing a circle


(f) drawing an anticlockwise circle


(g) tapping

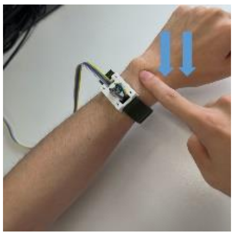

(h) double tapping

Figure 4.1: Appearances of the gesture

4.2 Data collection

4.2.1 Equipment

At the beginning, we decided to use a 3D-printed mockup smartwatch instead of concrete smartwatch due to its lower cost and ease of use in data collection. Figure

4.2 shows the mockup with a NATO type watch belt attached. The mockup has flat holes and rectangular holes which allow the belt and cables going through on both sides. Figure 4.3 shows the accelerometer (Kionix, KXR94-2050) fixed into a square hole which lies on the center of the mockup. Arduino Nano controller receives the signals from accelerometer and processes them, then sends the signals to a PC.

Figure 4.2: Mockup with the watch belt

Figure 4.3: Accelerometer fixed on the mockup

The signal output voltage range of the three axes of the accelerometer is 330-2970 mV, and the acceleration of ±2g can be detected (g is the gravitational acceleration, usually gets 9.80 m/s2). The positive direction of the output value of the three axes is shown in figure 4.3.

4.2.2 Method

Eight male university students aged 21–24 joined the experiment as participants. The participants were expected to wear the watch mockup, the controller recorded the 3-dimensional signals, including x, y, and z axes, from the accelerometer at a frequency of 100 Hz. As the participants performed the eight types of gestures, each repeated ten times according to the instructions, resulting in a dataset containing 720 samples.

4.2.3 Data processing

Figure 4.4 illustrates the raw data of a sliding right gesture obtained from the accelerometer, with 120 timesteps and three channels for the x, y, and z axes.

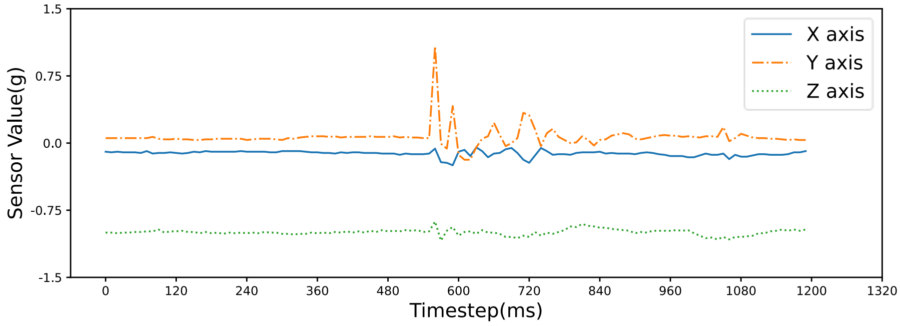


Figure 4.4: Raw data of a sliding right gesture

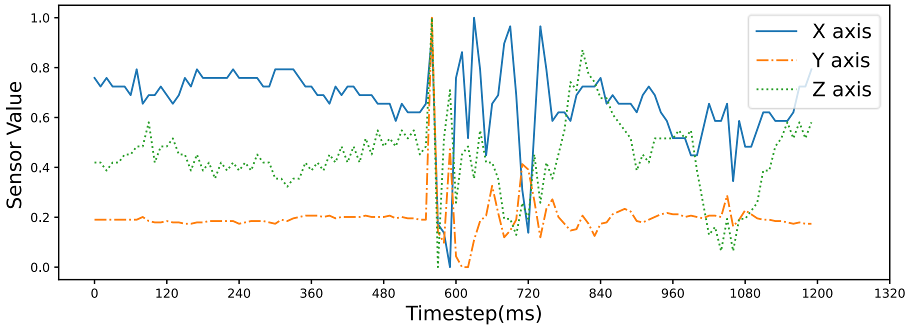


Figure 4.5: Standardized data of a sliding right gesture

At first, we considered the reasonability of the data without being denoised. we tried to use median and mean filters to process data and then train a neural network. We set three groups including no filter, median filter, and mean filter, however, after comparing the three groups of experiments, we find that the influence of noise on the results is exceedingly small.

$$Z = \frac{(X - \mu)}{\sigma} \tag{4.1}$$

Furthermore, we aimed to detect small waves generated by specific gestures, requiring us to amplify the waveform. For this purpose, the signals were standardized by z-score algorithm, in formula 4.1, Z represents the standardized data, X represents original data, μ represents the average data, and σ represents standard deviation.

Z-score algorithm proved to be beneficial in time-series data process. As shown in Figure 4.5, the small waves are enlarged.

4.3 Gesture independence Analysis

The paper designed a total of nine gestures, including keeping still gesture, with considerations based on previous gestures elicitation study. However, due to potential similarities in the effects on the accelerometer, meaning that different gestures may have similar signal waveforms, there could be cases where the recognizer struggles to accurately distinguish between them. To address this issue, we employed dimensionality reduction visualization using the features of these gestures to explore their similarities and determine the most feasible gestures.

Dimensionality reduction typically involves selecting an appropriate algorithm, where different algorithms may lead to varying results. Common linear dimensionality reduction algorithms include Principal Component Analysis (PCA) and Locally Linear Embedding (LLE) [48], while some scholars have organized neural network-based autoencoders [49], others are t-distributed Stochastic Neighbor Embedding (t-SNE) [50], Uniform Manifold Approximation and Projection (UMAP) [51]. These algorithms are unsupervised learning methods. However, the data collected in this study consists of time-series data with labeled information, making supervised learning methods more suitable. There are some supervised learning methods like Linear Discriminant Analysis (LDA). The first step in LDA involves calculating the within-class feature means. However, since the collected time-series data may have different vector distributions over time within the same class, the vector mean does not adequately represent the data features, making LDA unsuitable for direct use in this context.

Considering the characteristics of time-series data and drawing insights from various dimensionality reduction approaches, this thesis adopts a two-step approach for data reduction. The first step involves feature extraction, using the output of the convolutional recurrent neural network's hidden layer as the reduced feature. The second step is features dimensionality reduction, using the UMAP method to reduce the features to two dimensions, enabling intuitive clustering representation in a 2D graph.

Autoencoders are a type of neural network that learns to encode input data into a lower-dimensional representation and then reconstructs the original data from this representation, figure 4.6 shows the structure of it. They are used for tasks such as data compression, feature learning, anomaly detection, and generative modeling. By training on unlabeled data, they can extract meaningful features and patterns, making them valuable in various domains including image processing, natural language processing, and recommendation systems. Enlighted by autoencoder, this thesis proposes a feature extraction method, that is extracting features from hidden layers of neural networks. Autoencoder has the centrally symmetric structure, it can generate signals that fits the raw data, it means that the features represent the raw data well but may not be suitable for recognition.

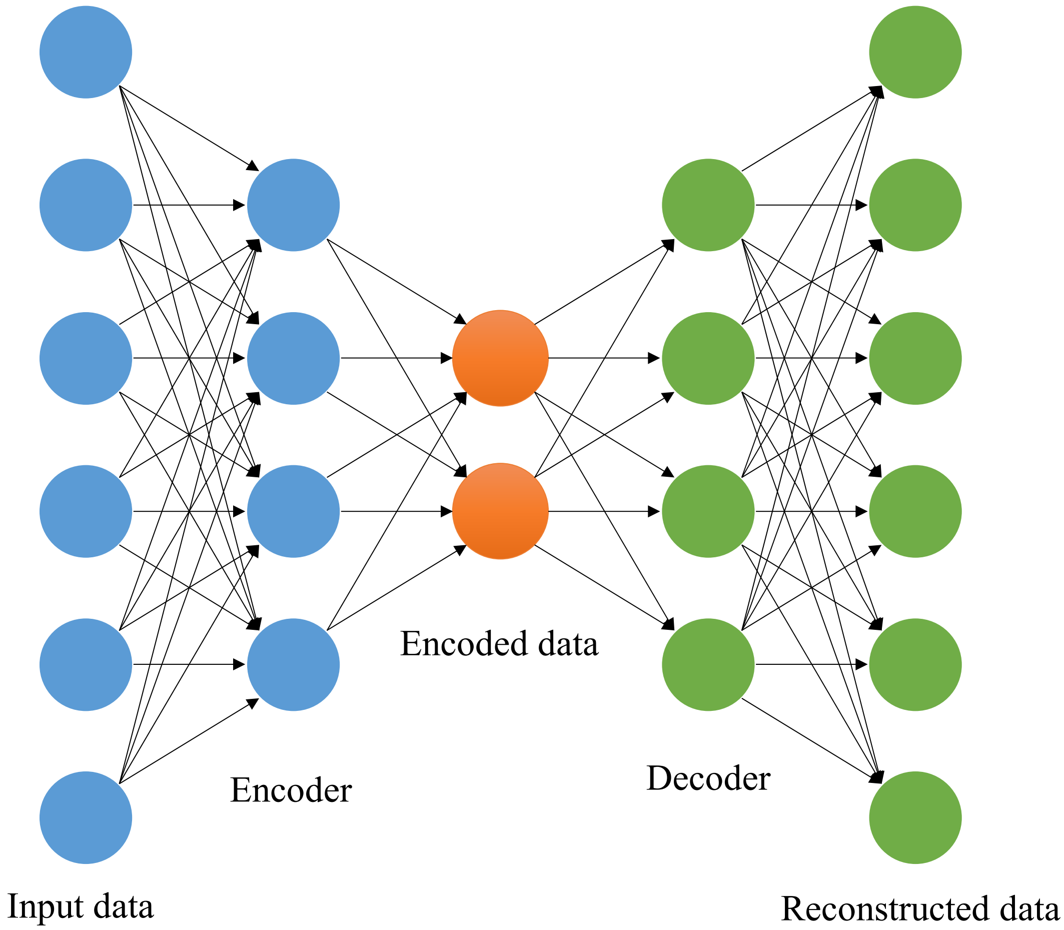


Figure 4.6: Structure of an autoencoder

A prior study demonstrated the utilization of RNN as a classifier for time-series data. We adapted the structure and added extra layers to satisfy our needs. Convolutional recurrent neural network (CRNN) combines CNN and RNN. It can handle variable-length sequence data and adaptively learn temporal and spatial features. The neural network employed for feature extraction is a CRNN, which undergoes multi-class recognition training using all sample data. The thirty output vectors from the final recurrent hidden layer serve as the sample features.

As this layer's output vectors directly yield the multi-class results through only one fully connected layer in the neural network, they can effectively represent the sample features once the network is professionally trained.

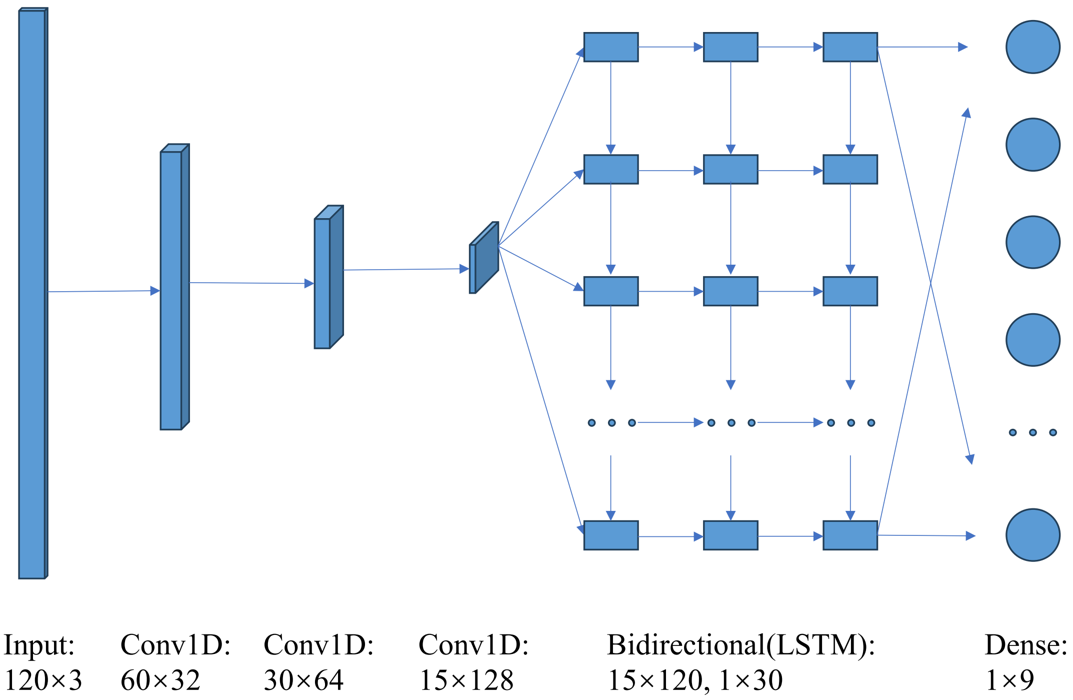


Figure 4.7: Structure of the CRNN

Figure 4.7 illustrates the structure of network. Before training the network for dimensionality reduction and clustering visualization, we preprocessed the input data.

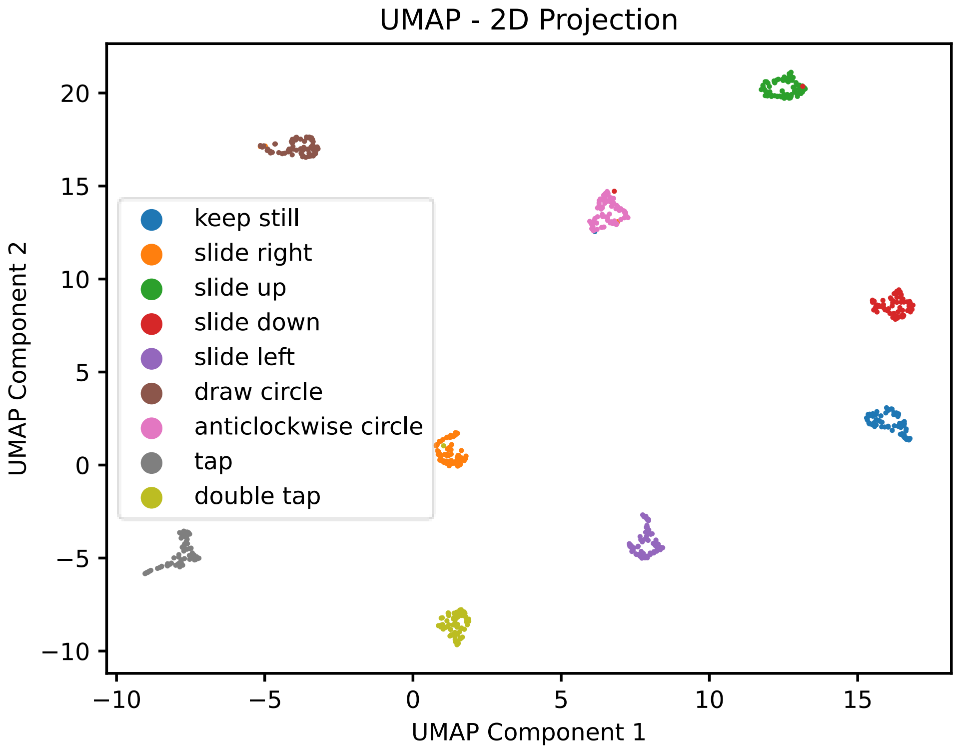


Figure 4.8: Result of UMAP

The UMAP method allows for preserving the global structure while mapping high-dimensional data to a low-dimensional space. In most cases, it achieves better efficiency and effectiveness than t-SNE. Figure 4.8 illustrates the dimensionality reduction effect graph of UMAP, it can be observed that most gestures can be correctly clustered together, with only a few exceptions. Therefore, the correlations between gestures are not significant enough to confuse the classifier, indicating that all nine gestures mentioned above are feasible.

4.4 Model training

As mentioned above, CRNN is used for feature extraction. In this thesis, we adapted the network, and use it for gestures recognition. Ensemble neural networks integrate multiple models to improve performance, reducing variance and overfitting, we adopt them to improve the accuracy. In our model, three networks with the identical architecture and average integration were employed, they have the same structure, but different random initial weights, figure 4.9 shows the structure of ensemble networks. This helps prevent the entire recognition model from falling into local optimality.

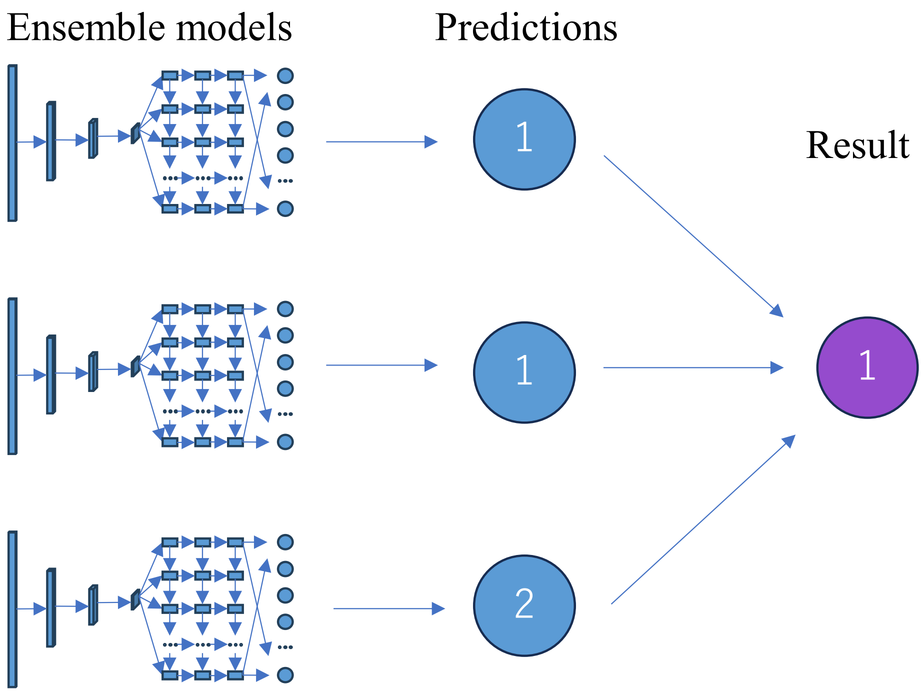


Figure 4.9: Ensemble networks

The activation functions are softmax for the output layer and linear rectification function (ReLU) for the other layers.

This thesis trained a single network to view the effect of parameters, the epoch number was set to 100, early stopping was enabled by default, batch size was automatically set to 8. Adam optimizer with 0.001 learning rate was used.

As shown in figure 4.10 and figure 4.11. The blue line represents the validation loss and accuracy, while the orange line represents the training loss and accuracy.

In these figures, validation loss started to get higher form 60 epochs, while accuracy reached the peak around 50 epochs. Thus, in the following network training, we choose 60 epochs as the training epoch number.

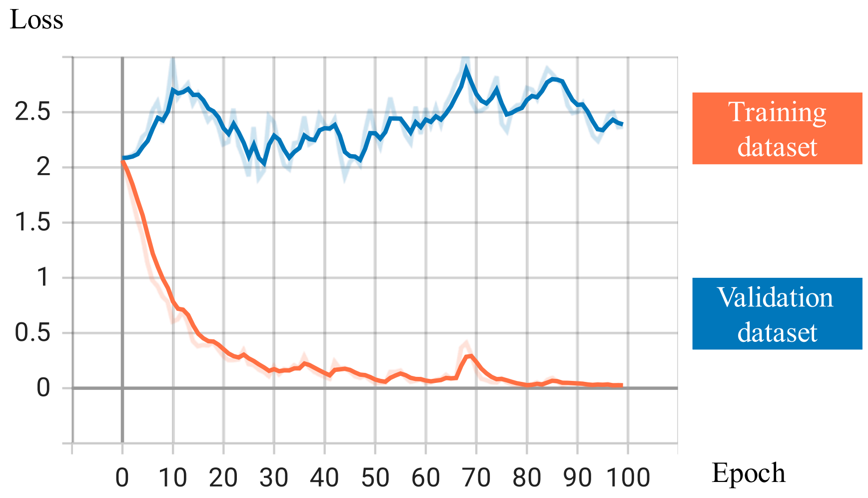


Figure 4.10: Training loss of single network

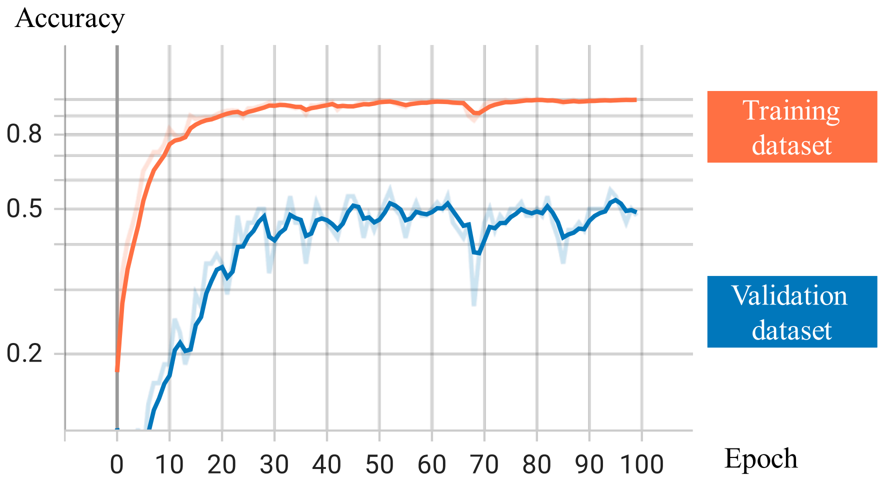


Figure 4.11: Training accuracy of single network

4.5 Result

4.5.1 Score measurement

To evaluate the model accuracy, 10-fold cross-validation was employed. We pooled data from all the participants and different gesture types and selected an equal number of each gesture type for the test dataset in each fold. In addition, this thesis chooses accuracy and F1-score as the indicators of model performance.

F1-score is a kind of indicator to measure the accuracy of binary classification model, considering the precision rate and recall rate of classification model. It is the harmonic average of the accuracy rate and the recall rate, with a maximum of 1 and a minimum of 0. To judge a multiple classification model, we used micro-f1, which counts True Positive (TP), False Positive (FP), False Negative (FN), True Negative (TN) of each category, and the sum is added to form new TP, FP, FN, and TN. Then Micro-Precision and Micro-Recall are calculated to get Micro-F1. Specifically, the confusion matrix of each category is calculated, and then the confusion matrix is "added" to get a multi-category confusion matrix, and then the F1score is calculated.

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.4}$$

$$F1 - score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{4.5}$$

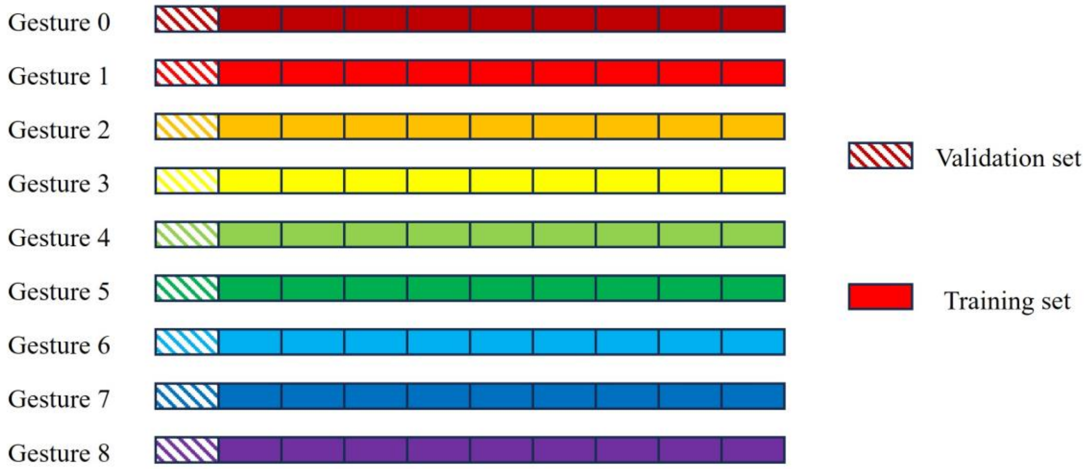


Figure 4.12: 10-fold cross validation

4.5.2 Around-smartwatch gestures

To confirm the recognition accuracy of the around-smartwatch gestures, a model was trained just for the four around- smartwatch gestures and the keeping still condition. As shown in Figure 4.11, the accuracy reached 85.69%, implying that it can recognize most gestures; however, further improvement is necessary before putting into application. Among these gestures, drawing circle had the lowest accuracy, with 22 out of 80 samples being incorrectly recognized. Of these misrecognized samples, 8 were classified as drawing anticlockwise circle. Besides, sliding left was another gesture with low accuracy, with most of the incorrectly recognized samples being classified as sliding right.

Table 4.2 shows their performance, except keeping still, tapping got the highest score, it is score reached 0.968, while double tapping was another gesture with high score. It means that these two gestures are promising in application scenarios. However, sliding right had the lowest score, its precision was only 67.78%, which means many other samples were incorrectly recognized as sliding right.

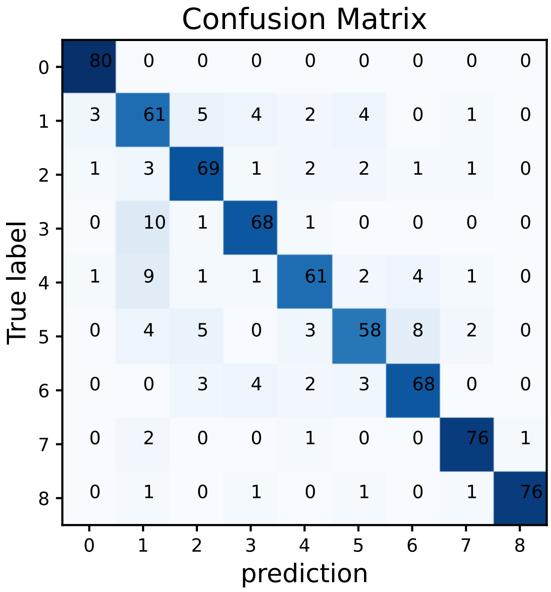


Figure 4.13: Confusion matrix

Table 4.2: Performance

Gestures	0	1	2	3	4
Name	keeping still	sliding right	sliding up	sliding down	sliding left
Accuracy	100%	76.25%	86.25%	85%	76.25%
Precision	94.12%	67.78%	82.14%	86.08%	84.72%
Recall	100%	76%	86%	85%	76%
F1-score	0.970	0.718	0.841	0.855	0.803

Gestures	5	6	7	8
Name	drawing a	drawing an	tapping	double tapping

	circle	anticlockwise circle		
Accuracy	72.50%	85%	95%	95%
Precision	82.86%	83.95%	92.68%	98.70%
Recall	73%	85%	95%	95%
F1-score	0.773	0.845	0.938	0.968

4.6 Discussion

Our model performs well in recognizing around-smartwatch gestures, which can be attributed to the integration of various network layers and data processing. However, the accuracy of some gestures was lower, which can be attributed to several possible factors.

Firstly, the signals of different gestures do not fluctuate in the same amplitude, and different gestures may have different waving axis. From Figures 4.14 and 4.15, we can see the difference between sliding right and tapping, sliding right gesture has only y axis fluctuating, but x and z axis of tapping gesture both fluctuates, which makes tapping easier to be recognized.

Secondly, during the repeats of one gesture, some signals were incorrectly collected, for example, performing a gesture too early before the accelerometer starting recording. Figure 4.16 illustrates one outlier in the repeats of the sliding right gestures; the recording did not cover the whole gesture duration. Figure 4.14 is the normal one.

Moreover, different participants had distinct action details, leading to variations in the signal waveforms. Figures 4.14 and 4.17 illustrate the same gesture from different participants.

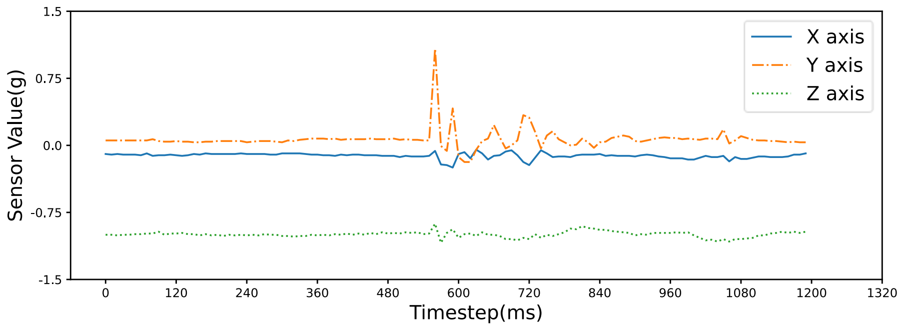


Figure 4.14: A signal of sliding right gesture from participant A

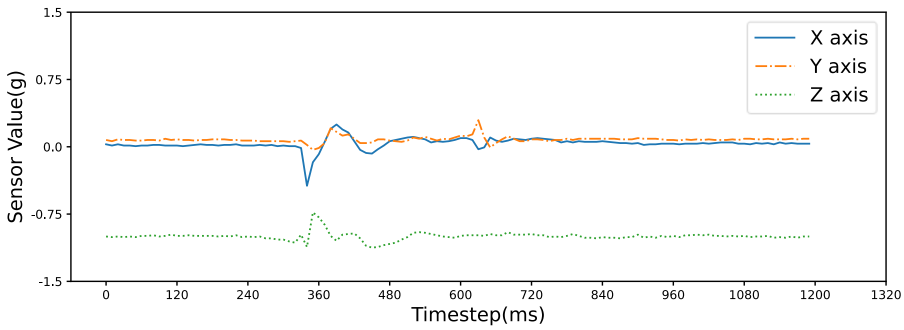


Figure 4.15: A signal of tapping gesture

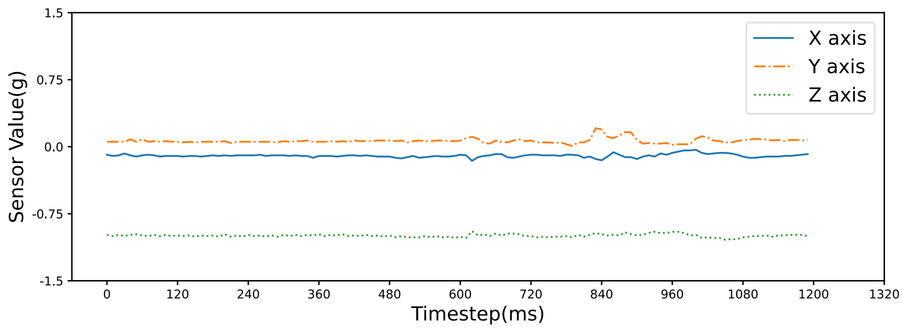


Figure 4.16: An abnormal signal of sliding right gesture

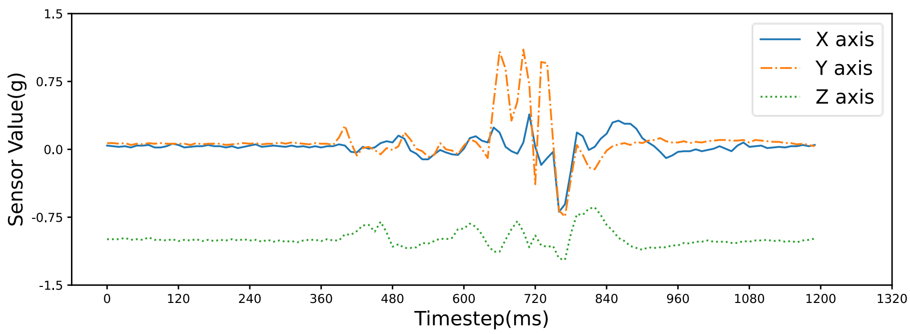


Figure 4.17: A signal of sliding right gesture from participant B

Chapter 5. Gestures recognition based in motion

In the third chapter, we conduct research on the elicitation of around-smartwatch gestures, and the music player is controlled in the state of motion. This set of gestures expands the way of human-computer interaction and has profound impact on the future human-computer interaction scenarios. In the fourth chapter, we collect the accelerometer data when the user performs around-smartwatch gestures in the static state and use the neural network to identify it. The practical feasibility of gesture recognition around-smartwatches is verified.

5.1 Collection system structure

We are extending the experimental scenario to simulate gesture recognition in motion rather than at still. Considering the noise of sensors in motion, our data collection experiments will be conducted on real smartwatches, and the gyroscope is used as an additional sensor. We have developed a set of Android applications. As the figure 5.1 shows, the client deployed on the smartwatch will collect data during the experiment and transfer it to the server deployed on the smartphone through Bluetooth connection. The smartphone will label and store the data. It is used for subsequent training and recognition tasks of neural networks.

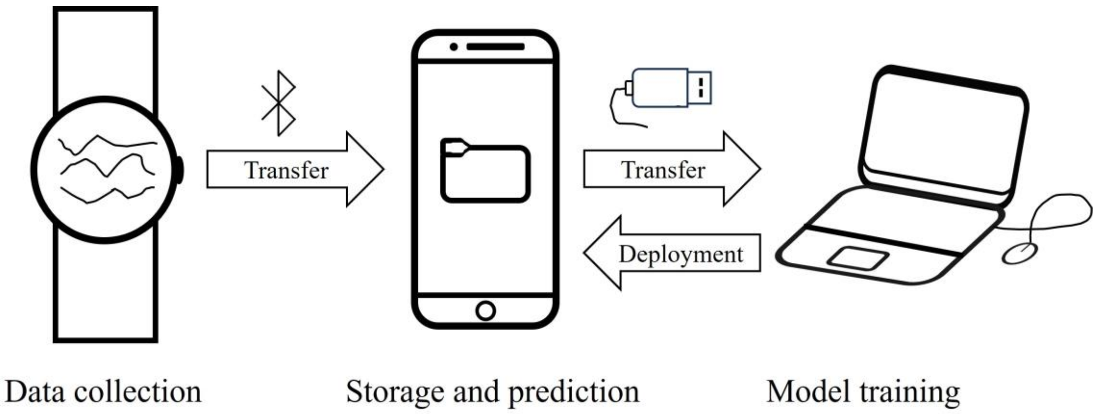


Figure 5.1: System structure

The development of both the smartwatch and mobile application is based on Android API 33, utilizing components from Android SDK and Android Jetpack, and developed using Android Studio.

The software deployed on the smartwatch functions as a client, comprising three main modules: interface, Bluetooth transmission, and sensor response. The interface includes three buttons—one for initializing the Bluetooth connection with the server, another for enabling sensor data collection, and a third for transmitting the collected data. Additionally, six text boxes are set up for visual monitoring of the sensor's operational status. In the Bluetooth module, a specific UUID is configured for Bluetooth socket connection with the server to recognize matching processes. Accelerometer and gyroscope listeners are set up, triggered upon the data collection button click, storing sensor data for 2 seconds in a temporary array, then enter a self-sleep mode. Figure 5.2 shows the interface on the smartwatch.

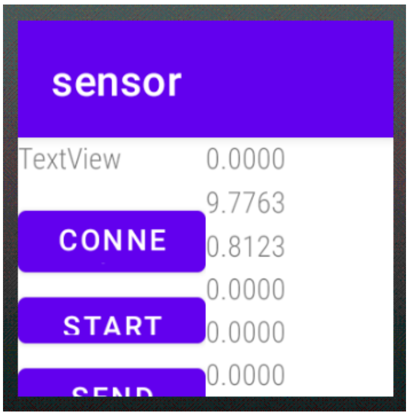


Figure 5.2: Interface on smartwatch

On the mobile application side, the deployed software functions as a server with three main modules: interface, Bluetooth transmission, and local data storage. The interface consists of three buttons—one for initiating the Bluetooth connection service, listening to client connection requests, another for reading float formatted data from the connected data stream and temporarily storing them in an array, and a third for storing the received batch of data in the smartphone's local external storage. Two text boxes are also present, one for recording the participant's name associated with the batch of data and the other for labeling the data. The Bluetooth module initiates upon the corresponding button click, opening the connection port and connecting when a process with a specific UUID requests access. It wraps the data stream class on the byte stream, enabling sequential parsing of floating-point numbers. The data storage module is used to label the collected data and store it in the corresponding folder under the experiment participant's name. Figure 5.3 shows the interface on the smart phone.

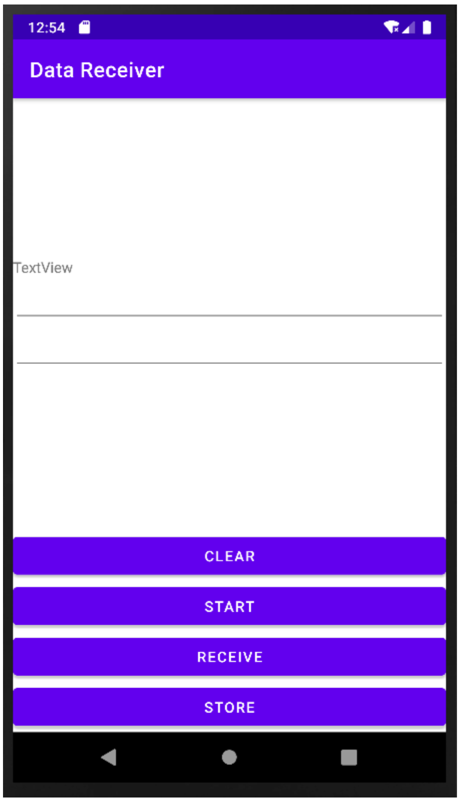


Figure 5.3: Interface on smart phone

5.2 Sample parameters

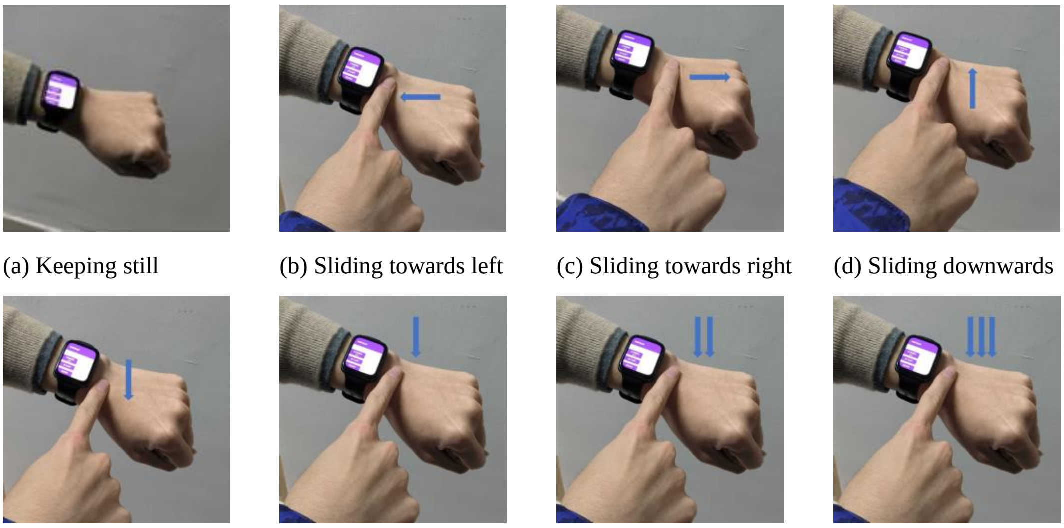
The device we use is an OPPO Smartwatch (OPPO Watch 2 42mm Bluetooth version, China), which uses Android Studio development tool and is developed based on Android 8.0 operating system. Each data acquisition time is set as 2 seconds, the acquisition interval is set as 10 milliseconds, the timestep is 200, and a total of six dimensions, including data from the three-axis accelerometer and three-axis gyroscope. Data is stored in Java float format. Six participants were invited to participate in the experiment, set up eight gestures including no action, each gesture was repeated 10 times, and a total of 480 groups of samples were collected.

5.3 Gestures set

As the gesture elicitation study in previous chapter shows, some gestures are assigned to corresponding commands while others are not, which means the recognition of left gestures is not necessary, we only collected the gestures which are assigned to commands. Here are seven gestures, corresponding to eight commands, and a control group of keeping still, they are sliding towards left, sliding towards right, sliding downwards, sliding upwards, tapping once, tapping twice, tapping for three times. Figure 5.4 illustrates all the gestures.

Table 5.1: Gestures set

No.	Name	Description
0	Keeping still	Do nothing but walk
1	Sliding towards left	Sliding on the right side towards left by a finger
2	Sliding towards right	Sliding on the right side towards right by a finger
3	Sliding downwards	Sliding on the right side downwards by a finger
4	Sliding upwards	Sliding on the right side upwards by a finger
5	Tapping once	Tapping once on the right side by a finger
6	Tapping twice	Tapping twice on the right side by a finger
7	Tapping for three times	Tapping for three times on the right side by a finger



(a) Keeping still (b) Sliding towards left (c) Sliding towards right (d) Sliding downwards

(e) Sliding upwards (f) Tapping once (g) Tapping twice (h) Tapping for three times

Figure 5.4: Gestures set

5.4 Result

5.4.1 Single network

As can be seen from Figure 5.5, the accuracy rate reached 50.6%. This is the result of 10-fold cross-validation after training with a single neural network. This result serves as a control group, after which a model composed of three integrated neural networks is trained for comparison.

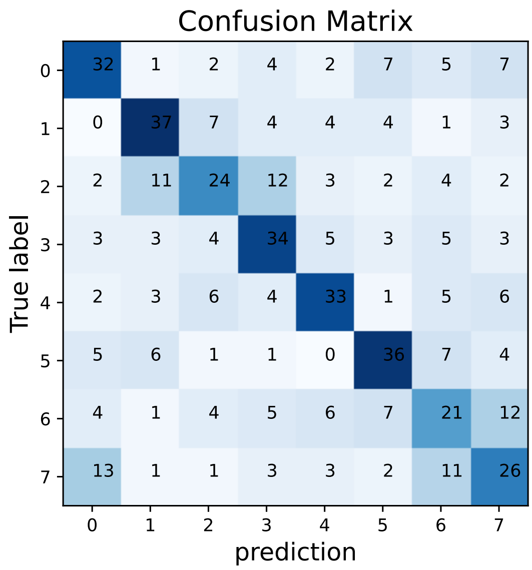


Figure 5.5: Confusion matrix of single network

5.4.2 Ensemble networks

Figure 5.6 shows the confusion matrix of the recognition model, the accuracy reached 56.88%, to compare it with the accuracy of single network in the last paragraph, this model is better. Thus, the advantage of ensemble model is shown here.

However, the accuracy is still lower than the experiment we conducted in chapter 4. The result is reasonable because this experiment was done under the situation when participants were moving.

In these gestures, sliding towards left gained the highest accuracy, which means that this gesture is practical in around-smartwatch gestures. While tapping twice gained the lowest accuracy, 10 out of 60 samples were incorrectly recognized as tapping for three times, besides, 14 out of 60 samples of tapping for three times were incorrectly recognized as keeping still. In this case, we consider alternative gestures for corresponding commands.

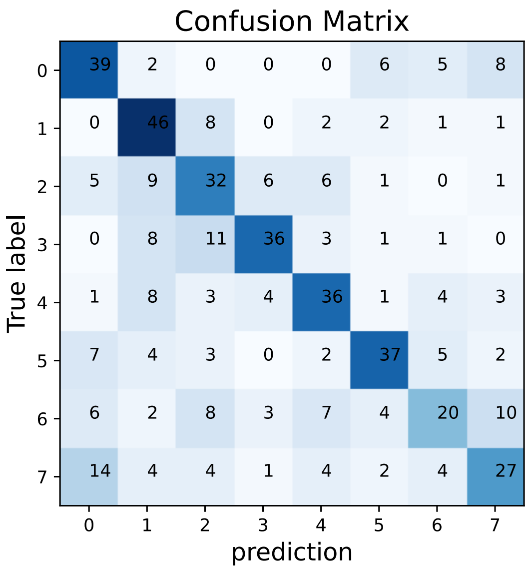


Figure 5.6: Confusion matrix of ensemble networks

5.5 Discussion

From the experimental results, we can see that the accuracy of the model has decreased significantly compared with the application scenarios in the previous chapter, from almost 90% to about 60%. Although it cannot be directly applied to smart watches at present, there is still the possibility of initial application after eliminating noise. This is the expected result of this thesis, because the accuracy of gesture recognition is affected by many factors.

The main problem is that in this experiment, the impact of ambient noise on gesture recognition is greater than in previous experiments. The figure 5.7 below is the accelerometer value collected in the experiment of Chapter 4 for a gesture that does not take any action while the user is still. The second 5.8 shows a signal of sliding toward right gesture in the experiment of Chapter 4. The figure 5.9 and figure 5.10 are the accelerometer and gyroscope values collected by the user in this experiment while walking at a slow speed without performing any gestures. And the other figures are the sliding towards right gesture while a participant walking at a slow speed.

In the first 5.7, we observe a remarkably stable signal, which serves as a control group and can be regarded as the noise of other gesture signals. Therefore, even in the presence of noise, it does not significantly interfere with the recognition of normal gestures. However, if we examine the figure 5.9 and figure 5.11 or figure 5.10 and figure 5.12 below, we notice significant fluctuations. Such fluctuations, when present as noise in other gestures, are likely to greatly affect the accuracy of gesture recognition. This is the primary reason for the low accuracy observed in this chapter's experiments.

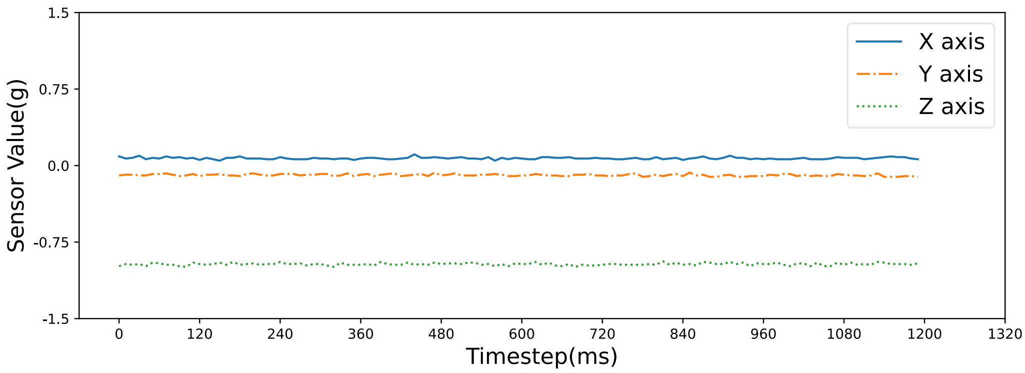


Figure 5.7: A accelerometer signal of no action at rest

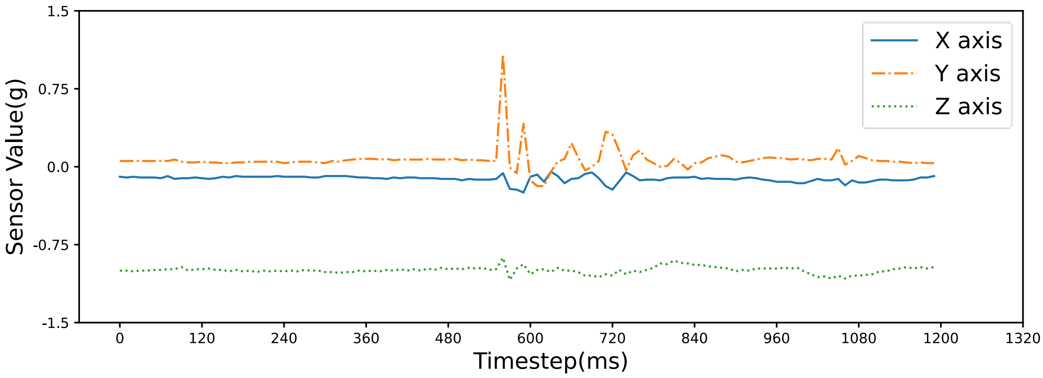


Figure 5.8: A accelerometer signal of sliding towards right at rest

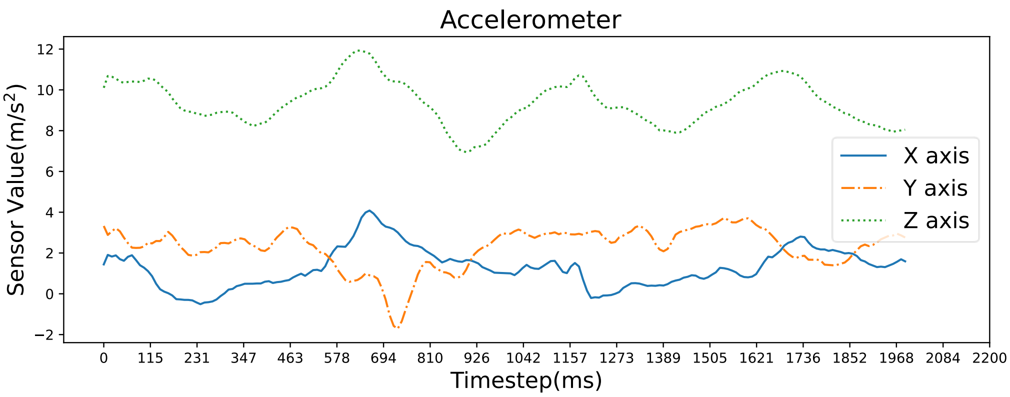


Figure 5.9: A accelerometer signal of no action while walking

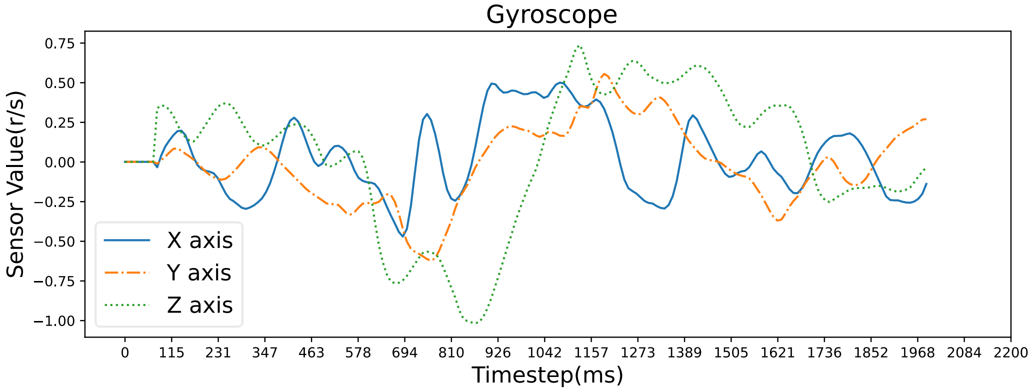


Figure 5.10: A gyroscope signal of no action while walking

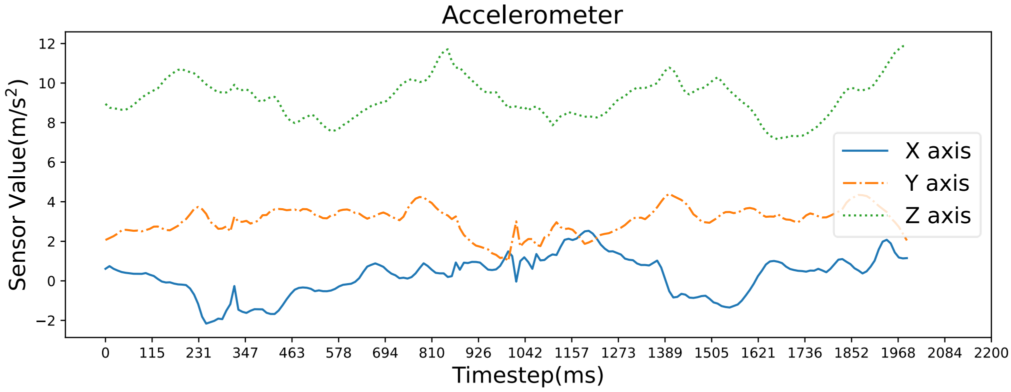


Figure 5.11: A accelerometer signal of sliding towards right while walking

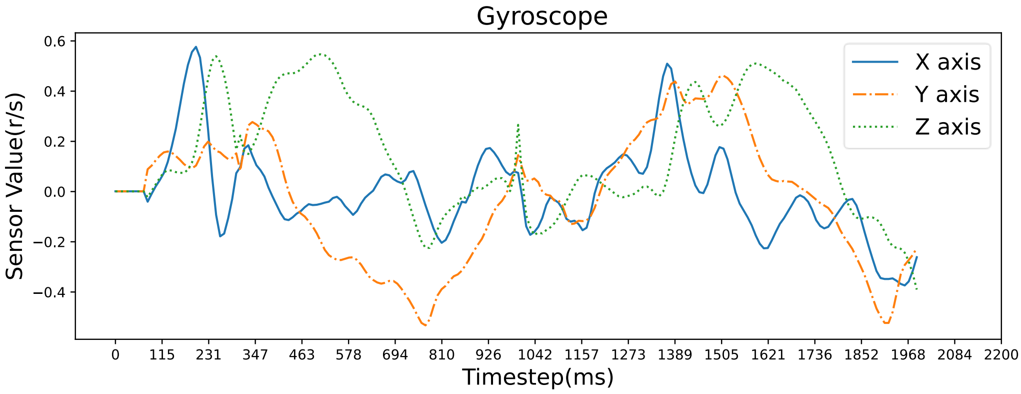


Figure 5.12: A gyroscope signal of sliding towards right while walking

Chapter 6. Conclusions

6.1 Research conclusion

In this thesis, we conducted a series of studies focusing on around-smartwatch gestures, progressing from research on around-smartwatch gestures elicitation study to utilizing accelerometers for recognizing the gestures. Finally, we extended the application scenarios by situating the operation of around-smartwatch gestures within the context of physical activities.

In the GES, participants invited to our experiments designed over twenty feasible around-smartwatch gestures, which, after categorization, were condensed into 18 subclasses. To integrate gesture operations with commands, we analyzed the vote counts and confidence score from a Likert scale for various gestures in different commands. High-scoring gestures were then combined with corresponding commands, forming a novel concept for a smartwatch interaction system.

Furthermore, to explore the feasibility of these gestures, we conducted a series of pilot experiments. An accelerometer was employed for the preliminary recognition of these gestures, and due to the implementation of neural network algorithms, our recognition system achieved high accuracy while reducing the hardware and software costs, enhancing the deploy ability feasibility of this technology compared to alternative methods.

Finally, we expanded the application scenarios of this novel type of human-computer interaction. Following the development of a visual interface for Android devices and the test of the neural network model, we designed a recognition system encompassing smartwatches, smartphones, and parallel computing devices. The system aimed to leverage collaborative efforts among intelligent devices to identify peripheral gestures on smartwatches in challenging motion scenarios.

These studies in this thesis establish a new pattern for human-computer interaction, capable of extending the channels for human-computer interaction and paving a feasible path for a more convenient technological lifestyle for humanity.

6.2 Future works

In the future research, this thesis will focus on the improvement of gesture recognition rate in motion state.

6.2.1 Scenarios extension

Although our research on gesture guidance and recognition is based on a few specific scenarios, such as the control of the headphone player in motion, the application can be used in other scenarios. Our findings can be applied to other situations, whether it is jogging, whether it is in a subway car, whether it's in a leaning position, and not just to music players.

For example, in another scene, in the hot summer, the employee drives home after work, he wants to turn on the intelligent air conditioning cooling in the home in advance, but after he gets on the car and fasten the seat belt, it is not convenient to turn on the mobile phone operation. At this time, he can raise his wrist and use around-smartwatch gestures to set the air conditioning mode, adjust the temperature fan and other operations. Figure 6.1 shows the commands and gestures.

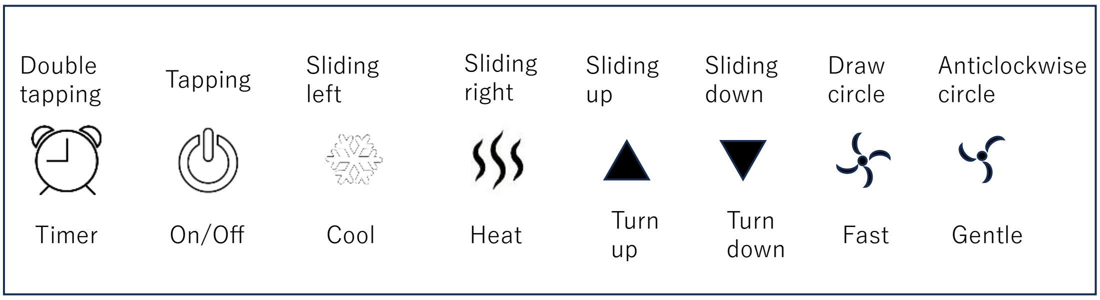


Figure 6.1: Another scenario of air conditioner control

6.2.2 Models' migration

Currently, our TensorFlow model is deployed and trained on a computer. In the comprehensive gesture recognition system, we plan to migrate the model to smartphones, while training will be conducted on cloud servers. When the smartwatch receives data, it will transmit the data back to the smartphone via Bluetooth connection. The neural network deployed on the smartphone will then recognize the signals, and the recognition results will be transmitted back to the smartwatch via Bluetooth. We trained the neural network model on a computer using CPU, and the time consumption remains within acceptable limits. Hence, we anticipate timely recognition on smartphones. The migration of this part of the model will be addressed as another future endeavor in this research.

References

[1] Smartwatches-Worldwide. Statista. 2024. Available: https://www.statista.com/outlook/hmo/digital-health/digital-fitness-well-being/fitness-trackers/smartwatches/worldwide

[2] John Koetsier: Global Smartwatch Market: Apple 34%, Huawei 8%, Samsung 8%, Fitbit 4.2%. Forbes. 2021. Available: https://www.forbes.com/sites/johnkoetsier/2021/05/27/global-smartwatch-market-apple-34-huawei-8-samsung-8-fitbit-42/

[3] Masoumian Hosseini, Mohsen, et al., “Smartwatches in healthcare medicine: assistance and monitoring; a scoping review,” *BMC Medical Informatics and Decision Making*, 2023, vol. 23, no.1, pp.248.

[4] Pay, Levent, et al., “Arrhythmias beyond atrial fibrillation detection using smartwatches: a systematic review,” *Anatolian journal of cardiology*, 2023, vol. 27, no.3, pp. 126.

[5] Kosch, Thomas, et al., “A survey on measuring cognitive workload in human-computer interaction,” *ACM Computing Surveys*, 2023, vol. 55, no. 13s, pp. 1–39.

[6] Xie, Yuzhen, Zihan Tang, and Aiguo Song, “Motion Simulation and Human–Computer Interaction System for Lunar Exploration,” *Applied Sciences*, 2022, vol. 12, no. 5: 2312.

[7] Lv, Zhihan, et al., “Deep learning for intelligent human–computer interaction,” *Applied Sciences*, 2022, vol. 12, no.22: 11457.

[8] Rawassizadeh, Reza, Blaine A. Price, and Marian Petre, “Wearables: Has the age of smartwatches finally arrived?” *Communications of the ACM*, 2014, vol. 58, no. 1, pp. 45–47.

[9] Yu, X., Zhou, Z., Xu, M., You, X., Li, X.-Y., “ThumbUp: Identification and authentication by smartwatch using simple hand gestures,” in Proceedings of *the*

2020 IEEE International Conference on Pervasive Computing and Communication, IEEE, New York, NY, USA, 2020, pp. 1–10.

[10] Wen, H., Rojas, J. R., Dey, A. K., “Serendipity: Finger gesture recognition using an off-the-shelf smartwatch,” in Proceedings of *the 2016 CHI Conference on Human Factors in Computing Systems*, ACM, New York, NY, USA, 2016, pp. 3847–3851.

[11] Zhou, J., Zhang, Y., Laput, G., Harrison, C., “AuraSense: Enabling expressive around-smartwatch interactions with electric field sensing,” in Proceedings of *the 29th Annual Symposium on User Interface Software and Technology*, ACM, New York, NY, USA, 2016, pp. 81–86.

[12] Ogata, M., Imai, M., “SkinWatch: skin gesture interaction for smart watch,” in Proceedings of *the 6th Augmented Human International Conference*, ACM, New York, NY, USA, 2015, pp. 21–24.

[13] Castro, Murilo Santos de, Pedro Raphael Inácio Gomes, and Thamer Horbylon Nascimento, “Integration of Smartwatches in Low-Cost Virtual Environments: A Systematic Literature Review with Emphasis on Cardboard Devices,” in Proceedings of *the 25th Symposium on Virtual and Augmented Reality*, ACM, Rio Grande, Brazil, 2023, pp. 279–283.

[14] Yan, Yu, Shota Yamabe, and Eric W. Cooper, “Method of Developing Hand Gesture Sets for Vehicle Entertainment and Ambiance Controls,” *2022 Joint 12th International Conference on Soft Computing and Intelligent Systems and 23rd International Symposium on Advanced Intelligent Systems*, IEEE, Ise-Shima, Mie prefecture, Japan, 2022, pp. 1–6.

[15] Ding Jr, Ing, Nai-Wei Zheng, and Meng-Chuan Hsieh, “Hand gesture intention-based identity recognition using various recognition strategies incorporated with VGG convolution neural network-extracted deep learning features,” *Journal of Intelligent & Fuzzy Systems*, 2021, vol. 40, no.4, pp. 7775–7788.

[16] Zimmerman, Thomas G., et al., “Applying electric field sensing to human-computer interfaces,” *Proceedings of the SIGCHI conference on Human*

factors in computing systems, ACM, 1995, pp.280–287.

[17] Kamachi, Haruka, et al., “Prediction of eating activity using smartwatch,” Adjunct Proceedings of *the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM International Symposium on Wearable Computers*, ACM, 2021, pp. 304–309.

[18] Santhalingam, Panneer Selvam, et al., “Synthetic smartwatch imu data generation from in-the-wild asl videos,” *Proceedings of the ACM on Interactive,* Mobile, Wearable and Ubiquitous Technologies, 2023, vol. 7, no. 2, pp. 1–34.

[19] Villarreal-Narvaez, Santiago, et al., “A systematic review of gesture elicitation studies: What can we learn from 216 studies?” In Proceedings of *the 2020 ACM designing interactive systems conference*, ACM, Eindhoven, Netherlands, 2020, pp. 855–872.

[20] Wobbrock, J.O., Aung, H.H., Rothrock, B. and Myers, B.A., “Maximizing the guessability of symbolic input,” in *CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05)*, ACM, New York, NY, USA, 2005, pp. 1869–1872.

[21] Lee, S.-S., Kim, S., Jin, B., Choi et al., “How users manipulate deformable displays as input devices,” in Proceedings of *the 28th International Conference on Human Factors in Computing Systems (CHI ’10)*, ACM, New York, NY, USA, 2010, pp. 1647–1656.

[22] Kerber, F., Löchtefeld, M., Krüger, A., McIntosh, J., McNeill, C., & Fraser, M., “Understanding Same-Side Interactions with Wrist-Worn Devices,” in Proceedings of *the 9th Nordic Conference on Human-Computer Interaction (NordiCHI ’16)*, ACM, New York, NY, USA, No. 28, 2016, pp. 1–10.

[23] Ruiz, J., Vogel, D., “Soft-Constraints to Reduce Legacy and Performance Bias to Elicit Whole-body Gestures with Low Arm Fatigue,” in Proceedings of *the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI ’15)*, ACM, New York, NY, USA, 2015, pp. 3347–3350.

[24] Tin Kam Ho, “Random decision forests,” in Proceedings of *3rd International Conference on Document Analysis and Recognition*, Montreal, QC, Canada, 1995,

vol. 1, pp. 278–282.

[25] Breiman, L., “Random Forests,” *Machine Learning*, 45, 2001, pp. 5-32.

[26] Rosenblatt, Frank, “The perceptron: a probabilistic model for information storage and organization in the brain,” *Psychological review*, 1958, vol. 65, no. 6, pp. 386.

[27] LeCun Y, Bengio Y., “Convolutional networks for images, speech, and time series,” *The handbook of brain theory and neural networks*, 1995, 3361.10: 1995.

[28] Amari S I., “Learning patterns and pattern sequences by self-organizing nets of threshold elements,” *IEEE Transactions on computers*, 1972, vol. 100, no. 11, pp. 1197–1206.

[29] Rumelhart D E, Hinton G E, Williams R J., “Learning representations by back-propagating errors,” *nature*, 1986, 323(6088), pp. 533–536.

[30] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” in *Neural Computation*, 1997, vol. 9, no. 8, pp. 1735–1780.

[31] H. Kim, H. Lee, J. Park, L. Paillat and S. -C. Kim., “Vehicle Control on an Uninstrumented Surface With an Off-the-Shelf Smartwatch,” in *IEEE Transactions on Intelligent Vehicles*, 2023, vol. 8, no. 5, pp. 3366–3374.

[32] Yao, S., Hu, S., Zhao, Y., Zhang, A., Abdelzaher, T., “DeepSense: A unified deep learning framework for time-series mobile sensing data processing,” in Proceedings of *the 26th International Conference on World Wide Web*, ACM, New York, NY, USA, 2017, pp. 351–360.

[33] Werbos, Paul, “Beyond regression: New tools for prediction and analysis in the behavioral sciences,” PhD thesis, Committee on Applied Mathematics, Harvard University, Cambridge, MA, 1974.

[34] Quinlan, J. Ross, “Induction of decision trees,” *Machine learning* 1, 1986, pp. 81–106.

[35] Arefin Shimon, Shaikh Shawon, et al., “Exploring non-touchscreen gestures for smartwatches,” in Proceedings of *the 2016 chi conference on human factors in computing systems*, ACM, San Jose, USA, 2016, pp.3822–3833.

[36] Gomes, Pedro Raphael Inácio, Murillo Santos de Castro, and Thamer Horbylon

Nascimento, “Gesture Recognition Methods Using Sensors Integrated into Smartwatches: Results of a Systematic Literature Review,” in Proceedings of *the XXII Brazilian Symposium on Human Factors in Computing Systems*, ACM, Diamantina, Brazil, 2023, pp. 1–11.

[37] Kwon, M.C., Park, G., Choi, S., “Smartwatch user interface implementation using CNN-based gesture pattern recognition,” *Sensors*, MDPI, Basel, Switzerland, 2018, vol. 18, no. 9, pp. 2997.

[38] Müller, Meinard, “Dynamic time warping,” *Information retrieval for music and motion*, 2007, pp. 69–84.

[39] Liu, Yilin, Fengyang Jiang, and Mahanth Gowda, “Finger gesture tracking for interactive applications: A pilot study with sign languages,” in Proceedings of *the ACM on Interactive, Mobile*, Wearable and Ubiquitous Technologies, 2020, vol. 4, no.3, pp.1–21.

[40] Chen, Y., Xue, Y., “A deep learning approach to human activity recognition based on single accelerometer,” in proceedings of *the 2015 IEEE International Conference on Systems, Man, Man, and Cybernetics*, IEEE, New York, NY, USA, 2015, pp. 1488–1492.

[41] Koch, P., Dreier, M., Maass, M., Böhme, M., Phan, H., Mertins, A., “A recurrent neural network for hand gesture recognition based on accelerometer data,” in proceedings of the 2019 Annual International Conference of *the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, New York, NY, USA, 2019, pp. 5088–5091.

[42] Bloch, V.; Frondelius, L.; Arcidiacono, C.; Mancino, M.; Pastell, M., “Development and Analysis of a CNN- and Transfer-Learning-Based Classification Model for Automated Dairy Cow Feeding Behavior Recognition from Accelerometer Data,” *Sensors*, 2023, vol. 23, no. 5, pp. 2611.

[43] Nguyen-Trong, Khanh, et al., “Gesture recognition using wearable sensors with bi-long short-term memory convolutional neural networks,” *IEEE Sensors Journal*, 2021, vol. 21, no.13, pp. 15065–15079.

[44] Wobbrock, J.O., Aung, H.H., Rothrock, B. and Myers, B.A., “Maximizing the

guessability of symbolic input," in *CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05)*, ACM, New York, NY, USA, 2005, pp. 1869–1872.

[45] Opportunity Dataset. 2012, Available: https://archive.ics.uci.edu/ml/datasets/opportunity+activity+recognition.

[46] Liu, Jiayang, et al., "uWave: Accelerometer-based personalized gesture recognition and its applications," *Pervasive and Mobile Computing*, 2009, vol. 5, no.6, pp. 657–675.

[47] Forster, Kilian, Daniel Roggen, and Gerhard Troster, "Unsupervised classifier self-calibration through repeated context occurences: Is there robustness against sensor displacement to gain?" *2009 international symposium on wearable computers*, IEEE, 2009, pp. 77–84.

[48] Sam T. Roweis, and Lawrence K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, 2020, 290, 5500, pp. 2323–2326.

[49] Walter Hugo Lopez Pinaya, Sandra Vieira, Rafael Garcia-Dias, and Andrea Mechelli, "Autoencoders," *Machine learning for data science handbook: data mining and knowledge discovery handbook*, 2023, pp. 353–374.

[50] Van der Maaten, Laurens, and Geoffrey Hinton, "Visualizing data using t-SNE," *Journal of machine learning research*, 2008, vol. 9, no. 11, pp. 2579–2605.

[51] McInnes, Leland, John Healy, and James Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," arXiv preprint arXiv, 2018, 1802.03426.