香港城市大學
City University of Hong Kong

Department of Computer Science

BSCCS Final Year Project 2021-2022
Interim Report I

21CS055

Gesture-Based Touch-Free smartwatch development with deep learning

(Volume __ of __)

Student Name	:	**HON, Hing Ting**
Student No.	:	**55776782**
Programme Code	:	**BSCEGU3**
Supervisor	:	**Dr XU, Weitao**
Date	:	**September 25, 2021**

For Official Use Only

Table of Contents

1. Introduction ... 3
   1.1. Motivation & Background Information ... 3
   1.2. Problem Statement ... 3
   1.3. Project Aims & Objective ... 4
   1.4. Project Scope & Deliverables ... 5
   1.5. Report Organization ... 5
2. Literature Review ... 6
   2.1. Review of Smartwatch Gesture Recognition Modalities ... 6
      2.1.1. Acoustic-based Gesture Recognition ... 6
      2.1.2. Sensor-based Gesture Recognition ... 6
   2.2. Review of Existing Gesture Recognition Algorithms ... 8
      2.2.1. Dynamic Time Warping (DTW) ... 8
      2.2.2. Support Vector Machines (SVM) ... 8
   2.3. Review of Deep Learning Classification Model ... 9
      2.3.1. Convolution Neural Network (CNN) ... 9
      2.3.2. Long Short-Term Memory (LSTM) ... 10
   2.4. Comparison & Discussion ... 10
3. System Design ... 12
   3.1. Overview ... 12
   3.2. System Diagram ... 12
   3.3. System Components ... 14
      3.3.1. Development Environment ... 14
      3.3.2. Gesture Vocabulary for Operational Functions ... 15
      3.3.3. Gesture Vocabulary for Text Entry ... 16
      3.3.4. Data Collection Component ... 17
      3.3.5. Classification Model Component ... 18
      3.3.6. Front-End Component ... 20
   3.4. Testing Procedures ... 21
      3.4.1. Experiment Setup ... 21
      3.4.2. Classification Model Testing ... 21
      3.4.3. System Testing ... 22
4. Plan & Schedule ... 23
   4.1. Project Schedule ... 23
   4.2. Gantt Chart ... 24
5. References ... 25
6. Appendix ... 28
   6.1. Monthly Logs ... 28

1. Introduction

1.1. Motivation & Background Information

Smartwatches have gained essential growth in popularity and have become ubiquitous in daily life by providing high mobility and user status-awareness that smartphones cannot achieve (Chun et al., 2018). Fingertip touching interaction is still the dominant scheme of smartwatch controlling approaches, which benefits from direct visual element interactions and inherits the existing usage from other touchscreen devices (Sun et al., 2017). Thus, smartwatches support efficient information access and response, including instant messaging and social networking through the enriched graphical interface on a small wrist-mount panel.

Different from smartphones, the high mobility of smartwatches provides a unique opportunity for recognizing physical activities with built-in inertial sensors. Wrist-mounted is the major characteristic that allows smartwatches to accurately collect physiological data through continual wrist connection (Rawassizadeh et al., 2014). To explore the potential usage of smartwatches, various machine learning models are integrated with the wearables to distinguish the specific arms and hand movements by classifying the movement pattern of wrist-mounted data (Mozaffari et al., 2020).

Recently, there has been an increasing awareness on utilizing wearables-based recognition to investigate the feasibility of providing immersive user experience and assisting digital device interaction with precise body language recognition (Poongodi et al., 2020; Kurz et al., 2021). Since the development of motion learning models is still in infancy, smartwatches with superior motion sensing and machine learning capability are the critical component in achieving the evolution of human-computer interaction (HCI) methods with explicit body language cognition.

1.2. Problem Statement

Despite the convenience and potential of smartwatches, surveys and literature investigations suggested that existing faulty interaction limits their usability by requiring precise target acquisition (Chun et al., 2018; Rawassizadeh et al., 2014). The small touch panel maintains smartwatch mobility but restricts the input and output capabilities. Touch interaction depending on tiny soft keyboard and buttons make tasks onerous and inefficient with two restrictions.

First, as shown in *Figure 1*, smartwatches cannot provide one-handedness reliably and restrict the usage scenario since the touchscreen is not accessible for the users whose non-wearing hand is not available or suffered from disabilities (Kurz et al., 2021). Additionally, smartwatches require users to operate with a tiny touch panel while the user's fingers can easily occlude over half of the watch face (Oney et al., 2013). Users frequently bent their index finger practically perpendicular to the panel to minimize finger occlusion and maximize visibility, but this usage habit imposes an additional burden on their hands (Hara et al., 2015).

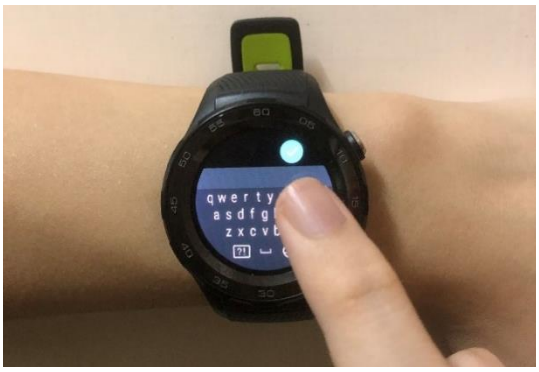


Figure 1. Illustration of finger occlusion for using smartwatch touchscreen

Alternative touch-free inputs modalities are necessary to mitigate those touch interaction shortcomings. Though hand motion recognition has the potential to expand interaction area from the restricted screen, it is still unclear how accurate and precise the hand motion in different scales can be recognized and leveraged to improve existing touch interaction deficiencies.

1.3. Project Aims & Objective

This project aims to employ deep learning models to recognize subtle hand movements for tasks operating on off-the-shelf smartwatches to address the above problem. The primary objective is to validate smartwatch hand motion recognitions performance in different preciseness, including arms, wrist, and finger motions. Rather than heavily relying on touch interactions, the suitable and feasible one-handedness interactions will be investigated to expand usage scenarios by leveraging different scales of single-hand gestures, including finger motions and finger writing with deep learning. The gesture-driven interaction is expected to enhance the user experience by reducing physical effort and optimizing visibility for commodity-level smartwatches.

1.4. Project Scope & Deliverables

This project's scope consists of developing a Wear OS application to collect the real-time motion data representing the designed gestures and hand motions from the embedded sensors with signal processing. The sensor-based deep learning model will be developed and trained with the collected dataset to recognize various in-air fine-grained gestures and index finger writing. The resulting model will be integrated with the smartwatch application to achieve a touch-free gesture-based interaction that requires less attention to perform and have symbolic meanings to map with ordinary operational functions and text entry.

1.5. Report Organization

The report is divided into three main sections. The first section explores the existing recognition techniques and background for extending smartwatch interaction. Then, the second section presents the detailed interaction scheme design, involved components, and implementation of the proposed solution. Finally, experiment results and demonstrations will explain how this proposed interaction scheme achieves the project objective.

2. Literature Review

This section reviews different existing research of gesture recognition modalities. In particular, the potential problems and improvements will be evaluated. Hence, various existing recognition algorithms and deep learning models will be analyzed to determine the feasibility and limitation of implementing touch-free interaction on the off-the-shelf smartwatch.

2.1. Review of Smartwatch Gesture Recognition Modalities

2.1.1. Acoustic-based Gesture Recognition

An acoustic signal is the sound waves or vibrations produced in response to specific activities. Recent acoustic sensing systems utilize embedded microphones and sensors from wearables to track the acoustic data reflected by gestures. Wang et al. (2020) applied acoustic signals to recognize finger gestures by classifying the measured wave's nuance with the microphones. Laput et al. (2016) developed an accelerated accelerometer data sampling to identify micro-vibrations of subtle hand motions propagating through the arm. Acoustic-based recognition can classify complicated gestures by differentiating the wave's nuance generated by various fine-grained finger motions. Nonetheless, acoustic signal transmission may experience dramatic signal distortion according to activity noise, so the maximum ambient noise level must be restricted during data capturing (Moreira et al., 2020). Therefore, acoustic sensing systems are not suitable for implementing passive gesture recognition in different environments.

2.1.2. Sensor-based Gesture Recognition

Smartwatch inertial sensors can track the slight wrist movements indirectly induced by hand gestures. When the user extends or bends a finger, the hand tendons will produce a subtle wrist movement, which is sufficient for classifying different hand gestures. Kurz et al. (2021) employed different inertial sensor data to classify hand sliding gestures with smartwatches and smart rings. Xu et al. (2015) also used the wrist and finger-mounted inertial sensors to recognize various arm, hand, and finger gestures. The Inertial Measurement Unit (IMU), especially accelerometer and gyroscope, are often leveraged to record the subtle gesture movement by monitoring the device movement in various research.

- Accelerometer data

An accelerometer is the standard sensor on modern smart devices to measure acceleration forces that represent the rate of body movement velocity. This sensor streams the waveform acceleration data in x, y, and z-axes to represent linear motions and the movement trajectory corresponding to the various wrist and arm activities by continuously contacting the users.

- Gyroscope data

A gyroscope is another embedded sensor to calculate a rotation describing the change of angular velocity over specific periods. This sensor captures the subtler angle change of tilt in three dimensions by spinning and aligning different orientations and positions to determine the movement difference. Rotational and angular velocity data sequences can represent a movement amplitude corresponding to the various small-scale motions pattern.

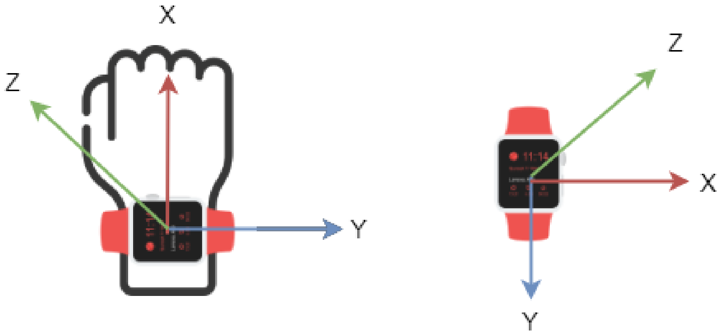


Figure 2.Illustration of smartwatch sensing movement directions

However, it is cautious that the sensor-based recognition is highly sensitive to arm orientation. Some unintentional subtle arm shaking might also produce a similar record, so most of the existing research relied on arms position fixation to prevent the arm-oriented noise. Hence, a learning algorithm is desired to achieve a case-sensitive gesture recognition experience by improving accelerometer-gyroscope data analysis with a larger dataset. A training dataset produced by multiple uses under various orientations is necessary for the algorithms to distinguish the waveform pattern difference between the designed gestures and the similar activities.

2.2. Review of Existing Gesture Recognition Algorithms

Recent research applied various pattern recognition and machine learning algorithms to process the accelerometer and gyroscope data combination to exhibit a superior accuracy in recognizing the gestures from off-the-shelf smartwatches.

2.2.1. Dynamic Time Warping (DTW)

Dynamic time warping (DTW) is the time analysis algorithms for measuring the similarity between two temporal data sequences with dynamic programming optimization. For gestures recognition, the DTW algorithm is often employed to evaluate the similarity between training and the input waveform sequence, which vary in performing speed for classifying the correct gesture pattern with minimum distance. Hamdy et al. (2014) suggested that DTW achieves high gestures recognition accuracy on user-dependent learning using accelerometer data template matching based on Euclidean distance. Yanay and Shmueli (2020) also proposed the combination of DTW and K Nearest Neighbors (KNN) to perform a user-dependent writing gesture recognition application by comparing new samples to the reference samples of a specific user.

However, DTW only has superior gesture recognition accuracy for the dataset related to the specific user by matching the sensor data sequences with high computation costs. This user-dependent recognition is not representative for all general users and cannot generalize for the user-independent paradigm. Thus, DTW is not suitable for implementing a robust gesture recognition that supports a more diverse range of unfamiliar users on smartwatches.

2.2.2. Support Vector Machines (SVM)

Support Vector Machines (SVM) is a supervised learning model that decomposes and classifies multi-dimensional data by discovering the most significant margin boundaries between different classes. SVM algorithm is widely employed for gesture motion recognition by classifying the extracted features from sensor data with lower computational complexity. Wen et al. (2016) introduced an SVM-based gesture recognition by training with the manually extracted features from the raw sensor data. The author suggested that the SVM model has outperformed accuracy over other existing machine learning algorithms, including K Nearest Neighbors (KNN), Naive Bayes (NB), and Logistic Regression (LR). Ameliasari et al. (2021) also employed SVM to implement a hand gesture detection with feature selection based on Pearson Correlation.

Nonetheless, most existing SVM-based method requires prior features extraction from the raw data, while this gesture classification accuracy often depends on the feature vector. It is required to convert the original data into representative data that the system can interpret with expert manual feature engineering and noise filtering. Data loss may occur during the feature engineering to reduce the recognition accuracy and increase the complexity of the algorithms (Muralidharan et al., 2021).

2.3. Review of Deep Learning Classification Model

Traditional pattern recognition and machine learning algorithms are restricted by their raw data processing ability, as converting the original data into representative data with manual data processing are crucial for those algorithms. In contrast, deep learning algorithms can automatically extract raw data features from data types, including images, audio, and video, for detection and classification without manual intervention.

2.3.1. Convolution Neural Network (CNN)

Convolution Neural Network (CNN) is a feed-forward neural network that is proved to have an outstanding recognition accuracy on analyzing spatial data with the concept of convolutions and pooling (Muralidharan et al., 2021). The convolutional layers detect and filter the features from the inputted spatial data, while the pooling layers repeatedly reduce the data dimensionality to distill the essential elements and reduce overfitting. It is further passed to fully connected layers for classification. Compared with SVM, the CNN model can directly interpret the original data without prior features extractions to learn the features automatically through the network and prevent data loss in the manual feature extraction process (Kwon et al., 2018).

Finger sliding gestures, which is also considered one type of HAR, can be classified using convolution to process the signal data. Kwon et al. (2018) and Yanay and Shmueli (2020) reported that the CNN model could recognize hand sliding gestures without being restricted by user dependency. Different from the DTW algorithm, CNN can perform a more accurate and refined gesture motion analysis for the small segments with large kernel sizes and network depth (Chu et al., 2020). Though 1D CNN can only process the sensor data in a fixed time length, it is relatively straightforward to train the model by learning high-level sensor data features of specific gesture motion under increased datasets.

2.3.2. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is an extended deep learning model of recurrent neural networks. It can exploit long-term dependencies with its complicated memory cell structure that feeds the results back to the network. LSTM retains critical data from the previous inputs and influences the current output. The memory cell leverages the gating concepts to memorize the previous data values over arbitrary time intervals and recognize the temporal correlations between sensor data samples for activities recognition (Zebin et al., 2018). Different from CNN, LSTM can classify the variable-length or infinite-length time-series data with an unknown delay period between critical activities. It reclusively extracts the sequential data according to specific sliding window size and maps it to a specific movement.

Due to its insensitivity to the delay period length, LSTM is ideal for classifying the long-term and variable sensor data sequence to solve the dynamic motion recognition with many-to-many inferencing. Tai et al. (2018) and Chu et al. (2020) applied LSTM algorithms to achieve sensor-based multiple continuous gestures recognition that is highly correlated with time series. Since LSTM has less feature extraction compatibility with many weights and biases parameters, the overall classification accuracy is worse than CNN, which focuses on extracting higher-level features in the highest computational complexity for a relatively short data input that is not highly correlated with time-sequence (Oluwalade et al., 2021).

2.4. Comparison & Discussion

Table 1. Comparison of existing research on smartwatch interaction

Solution	Gesture Presentation	Modalities	Classification	Feature Engineer	Perfor mance
Wang et al. (2020)	15 in-air finger motion gestures	Acoustic signal with microphone	CNN-LSTM	✗	98.4%
Laput et al. (2016)	17 in-air finger motion gestures	Bio-acoustic signal with accelerometer	SVM	✓	94.3%
Hamdy et al. (2014)	8 in-air hand sliding gestures	Accel	DTW, KNN, SVM, HMM	✓	~98%
Yanay and Shmueli (2020)	26 in-air writing gestures	Accel, Gyro	DTW-KNN	✓	89.2%
			CNN	✗	95.6%

Ameliasari et al. (2021)	8 in-air hand sliding gestures	Accel, Gyro	SVM	✓	94%
Wen et al. (2016)	5 in-air finger motion gestures	Accel, Linear Accel, Gyro	SVM, NB, LR, KNN	✓	98% (SVM)
Kurz et al. (2021)	12 in-air hand sliding gestures	Accel, Gyro	RF, SVM, KNN, NB	✓	98.8%
Kwon et al. (2018)	10 hand sliding gestures on surface	Accel	CNN	✗	97.3%
Chu et al. (2020)	11 in-air hand sliding gestures in sequences	Accel, Gyro	CNN	✗	92.3%
			LSTM	✗	86%
Tai et al. (2018)	6 in-air hand sliding gestures in sequences	Accel, Gyro	LTSM	✗	95%

Though the traditional pattern recognition algorithms, including SVM and DTW, are investigated to recognize hand gestures with relatively small datasets and few outliers effectively, the deep learning models, including CNN and LSTM, are expected to perform better classifications due to the capability of processing large dataset with powerful computational engines and automatic feature extraction under reduced loss. The deep learning model improves the gesture recognition accuracy in a user-independent manner for every scenario by dealing with complicated time series analysis of raw accelerometer and gyroscope training data produced by multiple users.

Recent research employed the deep learning model to implement gesture recognition. However, most of them only focus on the limited number of in-air hand sliding gestures and have no detailed implementation to apply the gestures recognition for improving smartwatches interactions. Hence, this project will develop an offline deep learning model to investigate the accuracy and feasibility of identifying more gesture categories and achieve the high robustness of touch-free smartwatch interaction. The prototype design and methodology will focus on leveraging the gesture-driven interaction with the in-air hand gestures to associate with fundamental operational tasks in ordinal smartwatches to achieve the project goals.

3. System Design

3.1. Overview

This project focuses on leveraging 15 in-air hand gestures for operational functions and 5 fingertip writing gestures for English characters entry to develop a touch-free smartwatch interaction scheme. The proposed application relies on a deep learning classification model to implement the above recognition techniques with embedded sensors from smartwatches. This system consists of three major technical components: sensor data collection that captures accelerometer and gyroscope data of hand motions propagated to wrist, a deep learning model that detects and classifies the designed hand gestures, and the front-end application that displays the visual elements interactable with designed gestures.

3.2. System Diagram

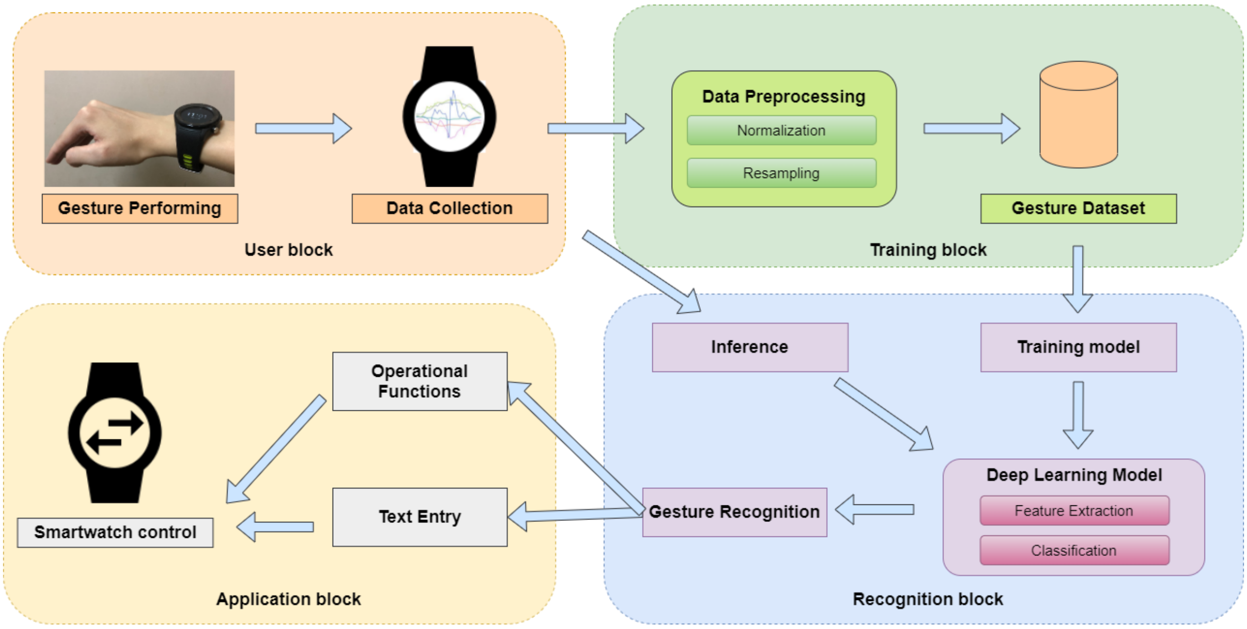


Figure 3. System diagram of the proposed application

Figure 3 illustrates all application components and the system architecture for the proposed gesture-based interaction system. The system is mainly divided into three blocks: training, recognition, and application. The application highly depends on the quantity and quality of user gesture inputs to provide gestures data samples and interact with the front-end application. For all the blocks, the sensor data collection will be handled by the smartwatch to ensure the shortest latency on logging the raw accelerometer and gyroscope data instead of transmitting it to external devices for analysis.

- Training Block

For the training stage, it is required to priorly collect a set of accelerometer and gyroscope data samples related to the designed gestures. After collecting sufficient training data from multiple users, “Normalization” and “Resampling” are the data preprocessing techniques that transform the sensor data with a specific fixed range and timestamp to remove the unnecessary data. The raw dataset with gesture class labeling will be leveraged to train the offline deep learning model for feature extractions and classification under the supervised learning approach.

- Recognition Block

The model will be integrated as the core in the backend systems of the smartwatch interaction application for the gesture recognition stage. The deep learning recognition model will track the real-time sensor data to identify the major two scales of designed gesture classes, including the in-air finger and fingertip writing gestures. The model will compute the most possible performing gesture by extracting the features and performing inference with the real-time captured user motion sample from the accelerometer and gyroscope sensors.

- Application Block

After the model computes the most possible performing gestures, the resulting gesture is treated as the primary input for the application. Each gesture class is mapped to the unique operations on the smartwatch application. When the specific gesture is detected, the system will execute the corresponding operational functions and English character entry command. The visual elements from the front-end application will have an instant response according to the performed gesture.

3.3. System Components

3.3.1. Development Environment

- Hardware and Software Development

This project aims to develop a robust touch-free interaction prototype on off-the-shelf smartwatches with sensor-based gesture recognition. Huawei Smartwatch 2 2018 operating Android Wear OS 2.1 is used as the major development platform for this project. The accelerometer and gyroscope are embedded on this smartwatch to capture the force and angular rate of different hand movements in a three-dimensional. Wear OS based on the Android platform is optimized for various series of commodity-level smartwatches and allows easier access to the hardware information. Android Studio will be the major development tool to develop the Wear OS applications with sensor motion data collection, gesture recognition, and gesture-based functions demonstration. This native application utilizes the Android APIs for sensors management, sensor data gathering, and input elements control with Java programming languages.

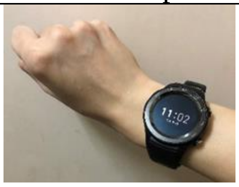
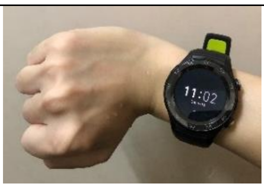
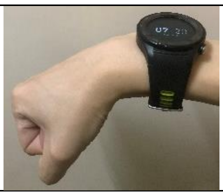
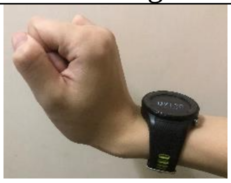
- Model Development Platform

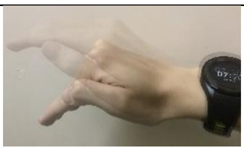
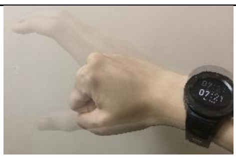
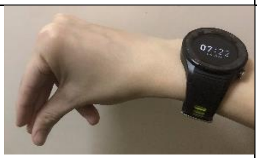
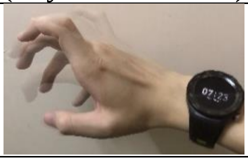
For the deep learning model development and the data preprocessing stage, TensorFlow is utilized for training a deep learning model to achieve modest parallelism with an open-source deep-learning library. Since Wear OS is the major development environment for our project, TensorFlow Lite expands the TensorFlow model's capability into a mobile environment, is suitable for implementing and optimizing the deep learning framework for on-device and IoT inference.

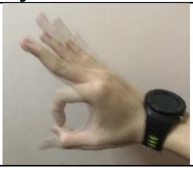
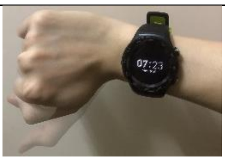
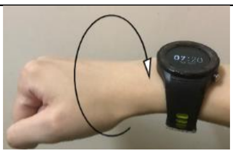
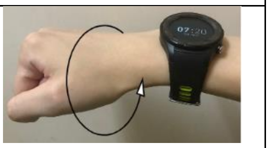
3.3.2. Gesture Vocabulary for Operational Functions

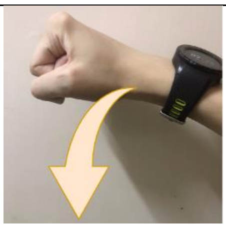
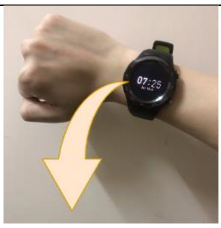
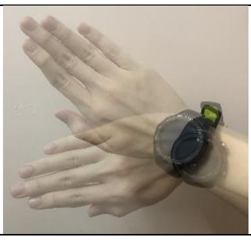
There are different operational functions required for operating smartwatches. To implement the low-effort interactions on smartwatches, 15 fine-grained gestures are designed with symbolic meanings and performed quickly and easily to memorize, so the designed gestures can avoid arm fatigue and require less attention compared with the large-scale gestures. *Table 2* shows all the supporting essential operational functions with the proposed gestures.

Table 2. Mapping of 15 operational functions and the gestures

Operations	Scroll up	Scroll down	Scroll left	Scroll right
Gesture Representation				
	Wrist Left	Writ Right	Writ Dropping	Writ Lifting

Operations	Select next item	Select previous item	Confirm selection	Back button (Physical button 1)
Gesture Representation				
	Finger Tapping	Hand Squeezing	Finger Snapping	Finger Waving

Operations	Exit button (Physical button 2)	Backspace	Back to top	Go to bottom
Gesture Representation				
	Finger Flicking	Thumb Rubbing	Arm clockwise rotation	Arm anti-clockwise rotation

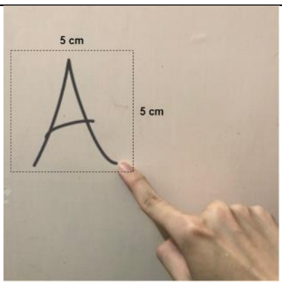
Operations	Undo	Redo	Clear
Gesture Representation			
	Vertical Arm Shaking	Horizontal Arm Shaking	Hand Sweeping

The above gestures are expected to be performed in two seconds, so the smartwatches can collect the high-frequency and low-amplitude movement data in fixed length and enhance sampling efficiency. It standardizes the data inference for the deep learning models with the fixed length of the sensor data and ensures that the user can complete the gestures without any burden.

3.3.3. Gesture Vocabulary for Text Entry

Besides the ordinary operational functions, the in-air fingertip writing gestures are proposed as the most intuitive way to perform text entry since handwriting can immediately reflect the human thinking of the desired character input. This project will only focus on recognizing the 5 in-air fingertip writing gestures and map to the first fifth upper-case English characters for the text entry demonstration. Table 3 shows all the supporting basic operational functions with the proposed writing gestures.

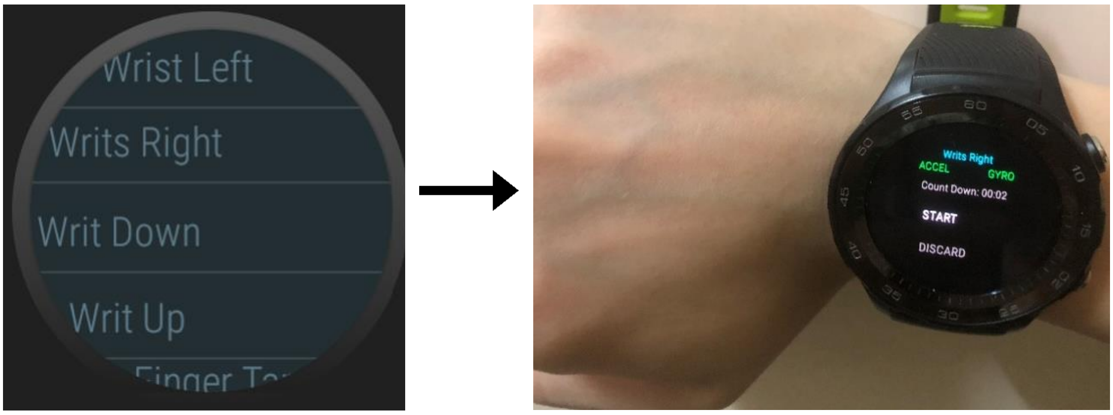
Table 3. Mapping of 5 English character entry and the finger pointing gestures

Operation	Touch Representation	Gesture Representation
5 Characters Text Entry (A to E)	 Typing on the soft keyboard	 In-air Writing

The above gestures are expected to be performed in variable-length since there are various strokes and writing speed to complete a specific English character with fingertip writing. Therefore, to align the recognition time and the writing range for each writing gesture, the maximum time to complete a writing gesture will also be set to two seconds inside an approximately 5cm * 5cm square area.

3.3.4. Data Collection Component

Since the sensor-based deep learning model is adopted for gesture recognition, a smartwatch data capture application is developed to collect the raw training data sequences from the embedded sensors, including the accelerometer and the gyroscope unit on the x, y, and z-axes at the sampling rate of 200 Hz. It is strictly required that the smartwatch need to be equipped on the wrist of the dominant hand to collect precise hand motions data sample of the above delicate motor gestures, so it is further assumed that the right hand is the dominant hand for this project, and all sensor data will be related to the right-hand gestures and writing.



Selecting target gesture class

Gesture performing for data collection

Figure 4. Illustration of Data Collection Flow

- Training Stage

This collector is used to priorly collect sufficient data samples with designed gesture class labeling to train the deep learning model. Moreover, the motion data of ordinary human activities, including typing, sitting, walking, and standing, will be collected to classify the null class that is recognized as no gesture performing. Due to the collection latency and computation overload concerns, the accelerometer and gyroscope data sequence will be saved as a CSV file instead of continuously transmitting the data to other devices or databases. As shown in Figure 4, it is required to select the current gesture class before starting the motion data capture. Then, the sensor data sampling function will be operated with a countdown strategy.

- Recognition Stage

After constructing the deep learning models, the data collector will be further integrated into the gesture-based interaction applications responsible for further collecting the raw sensory data to enrich the training dataset. To perform real-time gesture recognitions for interaction usage, the data collector will be automatically operated in the background to collect the accelerometer and the gyroscope sensor data to detect the hand motions continuously. Since all the gestures are designed to be completed in two seconds, the data collection operation will be triggered to capture the sensor data sample for two seconds interval iteratively.

3.3.5. Classification Model Component

Deep learning models will be proposed to perform the advanced gestures inference on the streaming accelerometer and gyroscope data obtained from the smartwatches. This project will adopt a lightweight one-dimensional CNN (1D-CNN) model to process the waveform pattern effectively. A lightweight 1D-CNN is superior at deriving the hidden signal features from shorter segments in fixed-length under smaller network configurations.

Since all gesture data will be captured under a fixed data length of two seconds, the fixed-length data will be more suitable to processed by the CNN algorithm rather than LSTM. Due to the advantages of rotational and positional invariance, it can reduce the overall loss of the raw sensor data to achieve high-level feature extractions. Since the proposed 1D CNN algorithm will interpret the fixed-length training data, the raw accelerometer and gyroscope data will be preprocessed with the normalization and resampling technique that transforms the sensor data with a specific fixed range and timestamp to fit the CNN model. The resulting data shape will be constructed by six time-series data channels (Ax, Ay, Az, Gx, Gy, Gz), represent the values of x, y, z-axis from the accelerometer and gyroscope unit according to the smartwatch position from a specific time step from 1 to T as shown as below.

$$Accelerometer\ data: Ax = \{a_x^1, \dots, a_x^T\}, Ay = \{a_y^1, \dots, a_y^T\}, Az = \{a_z^1, \dots, a_z^T\}$$

$$Gyroscope\ data: Gx = \{g_x^1, \dots, g_x^T\}, Gy = \{g_y^1, \dots, g_y^T\}, Gz = \{g_z^1, \dots, g_z^T\}$$

The gesture patterns are sampled at 200 Hz for two seconds, so the input data size of each axis is also fixed at 20, and the total input data size will be 2 * 200 * 6 from the 6-axis channel data regarding the accelerometer and gyroscope. The proposed 1D-CNN model is trained with a supervised approach by associating gesture classes with the corresponding sensor data sequence. It captures the temporal characteristics of inputted time series data to extract and learn the extracted features from sensor data by associating the window to a gesture pattern.

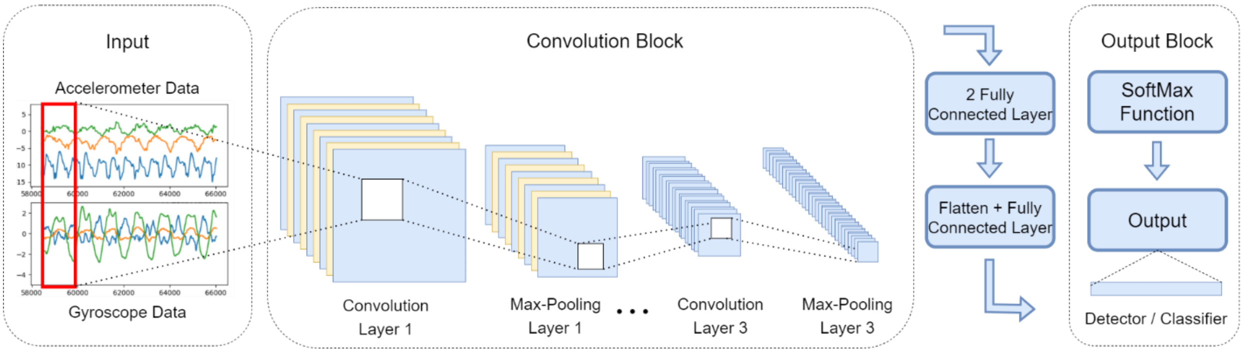


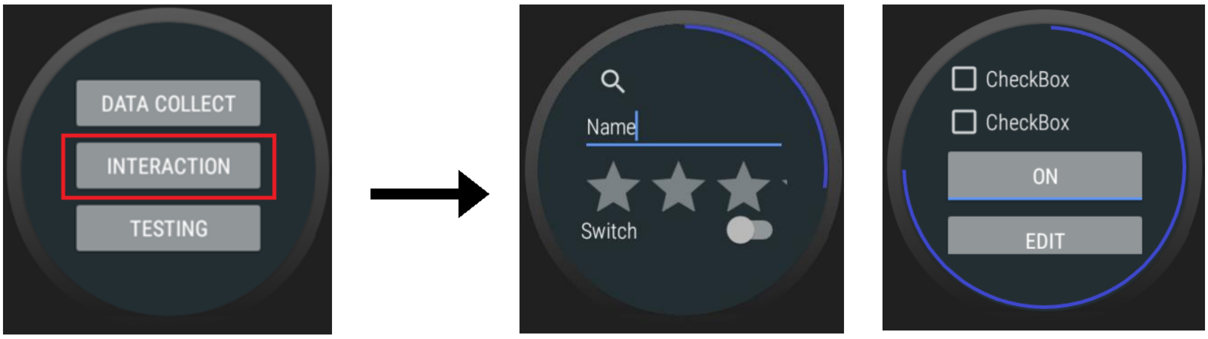
Figure 5. Semitic diagram of proposed 1D-CNN model

As shown in Figure 5, the designed CNN model will interpret the combinations of accelerometer and gyroscope data for each gesture class with the kernel that processes the segmented data across the entire sensor data sequence on the three 1D convolutional layers and max-pooling layers. The convolutional layers will continuously extract the convolved features from the data segments. The max-pooling layers perform signal maximizing for the sub-sampling region from the previous layer to obtain the abstract features that are represented compactly. It can also prevent overfitting and perform as a noise suppressant.

After that, there are three fully connected layers with various dense values (10, 20, 20) for combining the extracted feature output from the convolution block. The first fully connected layer will then stack all the learned features to produce a resulting individual representation of accelerometer and gyroscope data. The stacked features will be further imputed to another fully connected layer with a higher dense value for assembling all the extracted features into account. The following fully connected layer will then evaluate the temporal relationships by processing the flatten tensors of the temporal axis. Finally, the SoftMax operator will compute the probability of the specific gesture occurrence according to the evaluated flatten tensors for gesture detection and classification with the output vector sequence.

3.3.6. Front-End Component

To demonstrate how the gestures can be used to perform operation functions and text entry, the smartwatch music streaming applications will serve as an application example to display various common editable and interactable visual elements, including input boxes, buttons, scrollbar, and checkboxes. Most Wear OS applications support an extensive interface that provides a focused and scrollable view to display all items from the complete application in a list appearance. Hence, the operational functions, including scrolling, swiping, zooming, selecting, and texting are designed to browse the detailed content, select specific items, and edit the textboxes on smartwatches with the designed gestures. As shown in Figure 6, all the layout elements of the music streaming application will be adjusted to be controllable by the in-air gesture command and fingertip writing.



Home Navigation of the application

Different editable visual elements on extensive interface

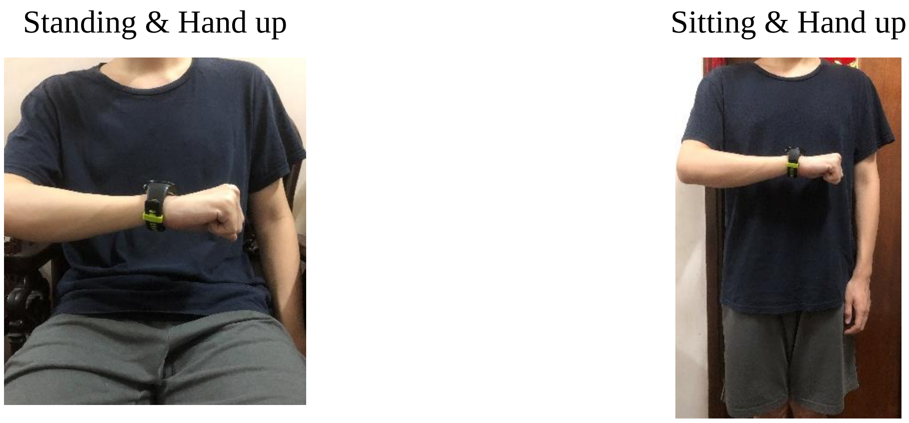
Figure 6. Illustration of Data Collection Flow

Since every hand motion is captured with an interval activation mechanism, the application will recursively collect and recognize the sensor data sample for each two-second interval. The two-second arc counting bar is displayed on the watch face periphery to notify the user about the starting point and the endpoint for the current sensor data collection. This design aims to prevent the data collection is overlapping in two distinct samples, while the user can perform the gesture at any point between the start and the stop event within the progress bar.

3.4. Testing Procedures

3.4.1. Experiment Setup

The data collector is leveraged to construct sufficient and comprehensive training dataset by collecting gesture samples from multiple users, which is also vital to evaluate the effectiveness and diversity of the classification models. Hence, this project will invite 6 participants to provide gesture data samples for the experiment. Participants are asked to repeatedly perform the assigned gestures for 20 segments under different behaviors and orientations, including standing, sitting, as shown in Figure 7. There are no restrictions on the positions of the devices, but it is required that the participants tightly wear the same smartwatch device on the wrist and perform the gestures with the right hand to guarantee the consistency of the training data.



Standing & Hand up

Sitting & Hand up

Figure 7. Data collection experiment under different orientations

After finishing the deep learning model and the recognition application development, the 6 sample providers and another new participant will be invited again to verify the correctness of gesture recognition functions. The reason of inviting new participants is to examine the capability of the proposed model to user-independent recognition for the unfamiliar gesture behaviors.

3.4.2. Classification Model Testing

It is necessary to verify the gesture recognition accuracy before launching it to the application. The preliminary data collected from the 6 participants will be split into training and testing data to verify the correctness of identifying the gesture pattern. The testing data is the subset of the dataset to measure the accuracy score on identifying all 20 designed gestures. To evaluate gesture recognition accuracy after model launching, each designed gesture will be performed ten times to collect the corresponding result. On the other hand, some ordinary human activities, especially typing on the keyboard and the similar gestures not included in this project, will be the false positive recognitions test cases to ensure that only the designed gestures can trigger the operations. The following equation will be calculated to evaluate the model recognition accuracy and precision with the test cases results.

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + True\ Negative + False\ Negative}$$

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

This project aims to obtain more than 90% accuracy and precision for recognizing all the 20 fingertip writing gestures with minimized false-positive recognition.

3.4.3. System Testing

The smartwatch system is integrated with the deep learning models to identify the gesture command for executing the corresponding operations and text entry. The system testing will verify the precision of identifying all the designed gestures as ordinary operation and character input. Since the application is expected to capture the gesture data within two seconds interval in the background, the recursive data sampling functions for every two seconds will be tested to ensure that the deep learning can receive and classify every real-time hand motion data when the application is activated.

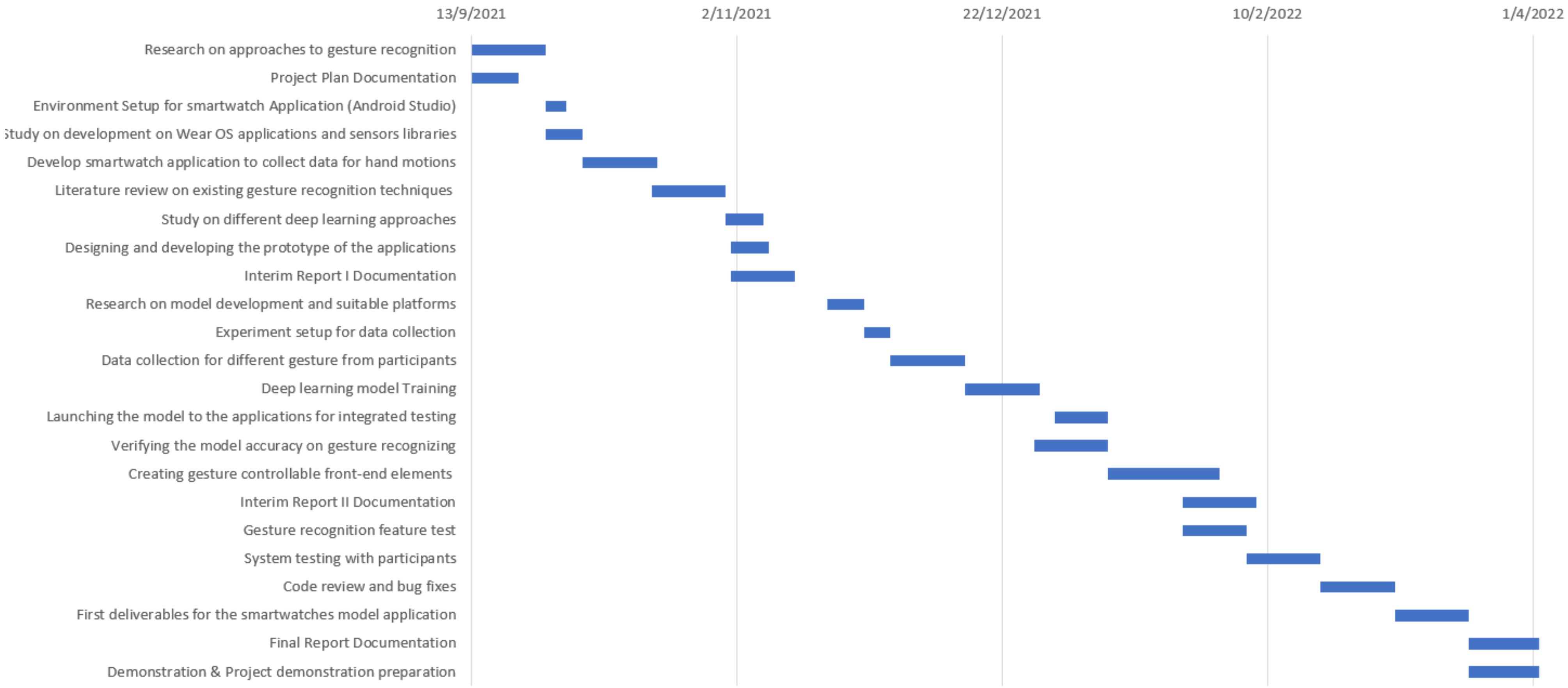
Moreover, the system testing approach also focuses on the interaction response and speed of front-end visual elements after receiving specific gestures input to ensure the smoothness of the gesture-based interaction. Hence, it is expected that all in-air gestures can be recognized without repeating to execute the corresponding controlling command instantly. Also, the system testing approach requires the testers to wear the smartwatch with the application activated and perform ordinary activities without performing any gestures for 15 minutes. It is expected that there are no responses or changes on the visual element since users do not perform any gestures to avoid false-positive recognition.

4. Plan & Schedule

4.1. Project Schedule

No.	Item	Expected Start Day	Expected Completion Date	Duration
1	Research on approaches to gesture recognition	13/9/2021	26/9/2021	14
2	Project Plan Documentation	13/9/2021	21/9/2021	9
3	Environment Setup for smartwatch Application (Android Studio)	27/9/2021	30/9/2021	4
4	Study on development on Wear OS applications and sensors libraries	27/9/2021	3/10/2021	7
5	Develop smartwatch application to collect data for hand motions	4/10/2021	17/10/2021	14
6	Literature review on existing gesture recognition techniques	17/10/2021	30/10/2021	14
7	Study on different deep learning approaches	31/10/2021	6/11/2021	7
8	Designing and developing the prototype of the applications	1/11/2021	7/11/2021	7
9	Interim Report I Documentation	1/11/2021	12/11/2021	12
10	Research on model development and suitable platforms	19/11/2021	25/11/2021	7
11	Experiment setup for data collection	26/11/2021	30/11/2021	5
12	Data collection for different gesture from participants	1/12/2021	14/12/2021	14
13	Deep learning model Training	15/12/2021	28/12/2021	14
14	Launching the model to the applications for integrated testing	1/1/2022	10/1/2022	10
15	Verifying the model accuracy on gesture recognizing	28/12/2021	10/1/2022	14
16	Creating gesture controllable front-end elements	11/1/2022	31/1/2022	21
17	Interim Report II Documentation	25/1/2022	7/2/2022	14
18	Gesture recognition feature test	25/1/2022	5/2/2022	12
19	System testing with participants	6/2/2022	19/2/2022	14
20	Code review and bug fixes	20/2/2022	5/3/2022	14
21	First deliverables for the smartwatches model application	6/3/2022	19/3/2022	14
22	Final Report Documentation	20/3/2022	2/4/2022	14
23	Demonstration & Project demonstration preparation	20/3/2022	2/4/2022	14

4.2. Gantt Chart



5. References

Hamdy Ali A., Atia A., Sami M. (2014). A comparative study of user dependent and independent accelerometer-based gesture recognition algorithms. In: Streitz N., Markopoulos P. (eds) *Distributed, Ambient, and Pervasive Interactions. DAPI 2014. Lecture Notes in Computer Science, vol 8530.* Springer, Cham. https://doi.org/10.1007/978-3-319-07788-8_12

Ameliasari, M., Putrada, A. G., & Pahlevi, R. R. (2021). An Evaluation of SVM in Hand Gesture Detection Using IMU-Based Smartwatches for Smart Lighting Control. *JURNAL INFOTEL*, 13(2), 47-53. https://doi.org/10.20895/infotel.v13i2.656

Chu, Y. C., Jhang, Y. J., Tai, T. M., & Hwang, W. J. (2020). Recognition of Hand Gesture Sequences by Accelerometers and Gyroscopes. *Applied Sciences*, 10(18), 6507. MDPI AG. http://dx.doi.org/10.3390/app10186507

Chun, J., Dey, A., Lee, K., & Kim, S. (2018). A qualitative study of smartwatch usage and its usability. *Human Factors and Ergonomics in Manufacturing & Service Industries*, 28(4), 186-199. https://doi.org/10.1002/hfm.20733

Hara K., Umezawa T., Osawa N. (2015) Effect of Button Size and Location When Pointing with Index Finger on Smartwatch. In: Kurosu M. (eds) *Human-Computer Interaction: Interaction Technologies. HCI 2015. Lecture Notes in Computer Science, vol 9170.* Springer, Cham. https://doi.org/10.1007/978-3-319-20916-6_16

Kurz, M., Gstoettner, R., & Sonnleitner, E. (2021). Smart Rings vs. Smartwatches: Utilizing Motion Sensors for Gesture Recognition. *Applied Sciences*, 11(5), 2015. MDPI AG. http://dx.doi.org/10.3390/app11052015

Kwon, M. C., Park, G., & Choi, S. (2018). Smartwatch User Interface Implementation Using CNN-Based Gesture Pattern Recognition. *Sensors*, 18(9), 2997. https://doi.org/10.3390/s18092997

Laput, G., Xiao, R., & Harrison, C. (2016). Viband: High-fidelity bio-acoustic sensing using commodity smartwatch accelerometers. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. https://doi.org/10.1145/2984511.2984582

Moreira, B. S., Perkusich, A., & Luiz, S. O. D. (2020). An Acoustic Sensing Gesture Recognition System Design Based on a Hidden Markov Model. *Sensors*, 20(17), 4803. MDPI AG. http://dx.doi.org/10.3390/s20174803

Mozaffari, N., Rezazadeh, J., Farahbakhsh, R., & Ayoade, J.O. (2020). IoT-based Activity Recognition with Machine Learning from Smartwatch. *International Journal of Wireless & Mobile Networks*, 12, 29-38. http://dx.doi.org/10.5121/ijwmn.2020.12103

Muralidharan, K., Ramesh, A., Rithvik, G., Reghunaath, A. A., Prem, S., & Gopinath, M. P. (2021). 1D Convolution approach to human activity recognition using sensor data and comparison with machine learning algorithms. *International Journal of Cognitive Computing in Engineering*. 2, 130-143. https://doi.org/10.1016/j.ijcce.2021.09.001

Oney, S., Harrison, C., Ogan, A., & Wiese, J. (2013). ZoomBoard: a diminutive qwerty soft keyboard using iterative zooming for ultra-small devices. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 2799-2802). https://doi.org/10.1145/2470654.2481387

Oluwalade, B., Neela, S., Gichoya, J.W., Adejumo, T., & Purkayastha, S. (2021). Human Activity Recognition using Deep Learning Models on Smartphones and Smartwatches Sensor Data. *HEALTHINF*. https://doi.org/10.5220/0010325906450650

Poongodi T., Krishnamurthi R., Indrakumari R., Suresh P., Balusamy B. (2020) Wearable Devices and IoT. In: Balas V., Solanki V., Kumar R., Ahad M. (eds) A Handbook of Internet of Things in Biomedical and Cyber Physical System. Intelligent Systems Reference Library, vol 165. Springer, Cham. https://doi.org/10.1007/978-3-030-23983-1_10

Rawassizadeh, R., Price, B. A., & Petre, M. (2014). Wearables: Has the age of smartwatches finally arrived?. *Communications of the ACM*, 58(1), 45-47. https://doi.org/10.1145/2629633

Sun, K., Wang, Y., Yu, C., Yan, Y., Wen, H., & Shi, Y. (2017). Float: one-handed and touch-free target selection on smartwatches. *Proceedings of the 2017 chi conference on human factors in computing systems* (pp. 692-704). https://doi.org/10.1145/3025453.3026027

Tai, T. M., Jhang, Y. J., Liao, Z. W., Teng, K. C., & Hwang, W. J. (2018). Sensor-based continuous hand gesture recognition by long short-term memory. *IEEE sensors letters*, 2(3), 1-4. https://doi.org/10.1109/LSENS.2018.2864963

Wang, Y., Shen, J., & Zheng, Y. (2020). Push the Limit of Acoustic Gesture Recognition. *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, 566-575. https://doi.org/10.1109/TMC.2020.3032278

Wen, H., Ramos Rojas, J., & Dey, A. K. (2016). Serendipity: Finger gesture recognition using an off-the-shelf smartwatch. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (pp. 3847-3851). https://doi.org/10.1145/2858036.2858466

Yanay, T., & Shmueli, E. (2020). Air-writing recognition using smart-bands. *Pervasive and Mobile Computing*, 66, 101183. https://doi.org/10.1016/j.pmcj.2020.101183

Xu, C., Pathak, P. H., & Mohapatra, P. (2015). Finger-writing with smartwatch: A case for finger and hand gesture recognition using smartwatch. *Proceedings of the 16th International*

Workshop on Mobile Computing Systems and Applications (pp. 9-14). https://doi.org/10.1145/2699343.2699350

Zebin, T., Sperrin, M., Peek, N., & Casson, A. (2018). Human activity recognition from inertial sensor time-series using batch normalized deep LSTM recurrent networks. In *IEEE EMBC*. https://doi.org/10.1109/embc.2018.8513115

6. Appendix

6.1. Monthly Logs

October 2021

- Complete the submission of Project Plan
- Complete the studies related to smartwatches sensors and the basic WearOS application development guide
- Complete the Literature review on the existing touch-free interactions scheme for smartwatches
- Focusing on the studies and research related to machine learning and deep learning algorithms that related to sensor-based gestures and human activities recognition