Personalized Exercise Training Chatbot based on Wearable Fitness devices

by

Zhiqiang Xiong

M.E., Huazhong University of Science and Technology, China, 2016

B.E., Wuhan University of Technology, China, 2014

A REPORT

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Computer Science
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2018

Approved by:

Major Professor
William Hsu

Copyright

Abstract

This report presents a personalized exercise training chatbot for individual users based on data collected from the Internet of Things (IoT), particularly wearable fitness devices. The chatbot is designed with our goal of motivating users to exercise more by discussing exercise statistics with the user, such as whether their daily steps have increased, decreased, or remained steady.

In this work I first survey a few examples of how increased interest in fitness and the promotion of healthy lifestyles is driving demand for personalized artificial intelligence, wearable computing, and ubiquitous computing applications. Next, I describe the design of a data-driven "personal trainer" chatbot. I then develop a prototype persuasion system based on interactive dialogs delivered via a front-end application, that collects data from wearable equipment using back-end data loggers that I instrumented as a mobile application. Finally, I describe the process of deploying and demonstrating this prototype along with technical challenges and early findings.

The overall system consists of (1) the back-end *Coach* agent, an Android application that collects data from all wearable instruments, and (2) the front-end *Me* agent, which initiates and continues conversations with the user using notifications that are in turn based on data from the *Coach* agent. This data-driven ensemble reminds the user to exercise and also gives the user a chance to provide feedback via human/agent interactive dialogs. In this publication, I used only one wearable device, the *MI Band 2*, and get real-time steps and weekly step aggregates from it. The human/agent dialogues are deployed via the Slack groupware platform. *Google Sheets* is used as a web service for updating and exchanging data.

Table of Contents

List of Figures

List of Tables

List of Listings

Acknowledgments

I would like to express my sincere gratitude to my Major Professor, Dr. William Hsu for his constant support and for trusting in my ability to complete this publication. He encouraged me all the way, especially at the end of publication. Many thanks to him.

I am also thankful for my committee members Dr. Doina Caragea and Dr. Arslan Munir for their encouragement, and time they spent serving on my committee.

My special gratitude goes out to my friend Irving Segovia for believing in me and supporting all the way in this research. I will miss the time when he video call to encourage me keep working even though it is hard some time. I will miss the joy whenever we had a even little progress. Hope we will keep in touch this way for the rest of our lives.

I would also like to acknowledge the support of ShanShan Wu by providing support and knowledge about Bluetooth connection for this publication. The time when we tried to review the Java code for extracting data from Xiaomi MI Band 2 always come across my mind when I read the data from Xiaomi MI Band 2. It is impossible to read data successfully without her help.

In the end, I would like to thank my family for their immense love and belief. It is great having them in my life. I will always be grateful to them.

Chapter 1

Introduction

Over the past decade, we have seen a rise in technologies targeting the promotion of a healthy lifestyle.[1] Electronic health and fitness trackers have received substantial attention recently, from new mobile and wearable technologies to evaluations of potential health impacts.[2] In addition to commercial devices such as the FitBit and Nike Fuelband, research projects have targeted areas such as food habits[3], physical activity[4], and mental well-being[5]. Therefore, it is common for individuals to have tried using mobile phones to track physical activity.[6] These trackers maybe give access about physical activity data from the user. However, it is hard for many people to keep tracking the physical activity data and improve it for a long term. Thus, encouraging people to exercise more is key to maintaining or regaining personal health but, unfortunately, difficult to achieve in practice. One barrier to exercising is that laypeople, that is, non-professional athletes often are insufficiently knowledgeable about effective and safe physical exercises. Maintaining a long-term exercise regime requires high levels of motivation and time demand, which often conflicts with people's busy lifestyles. It is well established that access to a personal trainer has a significant impact on both adherence to a physical exercise program, and the quality of the exercise undertaken.[7] Personal trainers continuously monitor the exercises and both provide individualized advice and motivate the trainee. They also play an important role in rehabilitation, e.g. exercise programs for muscle recovery after surgery, where the need for advice regarding effectiveness and safety is even

greater.[8]

Unfortunately, the availability of specialized personal body trainers is limited and in most cases simply too expensive to provide over extended periods of time. What's more, personal body trainers are impossible to be in touch with at anytime or place that the user might need. It is also hard for personal body trainers to collect all physical activities to motivate the users, or make suggestions and recommendations to them, in an appropriate way.

Even though many mobile applications of wearable fitness device notify the user to exercise more, but imagine a user is in a class or working and can only exercise afterwards; this user would have to close the notification. However, the user might forget to exercise after that if the application does not notify the user again. As another example, an application on a wearable fitness device notifies the user multiple times when the user's legs are hurt that week. The situations show that the persuasion system based on the wearable fitness device is too general, it is not user-adapted or personalized. Personalization plays an important role in this, as the most effective persuasive and motivational strategies are likely to depend on the user characteristics such as the user's personality, affective states, behaviors, knowledge, and goals.[9] User-adaptive and personalized are ways or models that can be adaptive based on the user's physical state and feedback.[10] For the same user, the same exercise and the same level, the users experience with the exercise is updated differently based on the different users heart rates during the exercise.

In order to solve the barrier mentioned above, we present a personalized exercise training chatbot based on wearable fitness devices to motivate the user exercise more. It is an Android phone-based prototype persuasion system based on human-agent dialogs and wearable fitness devices persuasion. The personalized exercise training chatbot not only persuades the users to exercise, but also gives the users the opportunity to state their situation and allows delay or cancel exercise plan if necessary. In this prototype persuasion system, we use only single wearable fitness device, the Xiaomi MI Band 2 and focus on walking activity. Thus, the whole system contains three parts, the Coach agent, the Me agent and the database. First, the Coach agent includes an Android mobile application and Xiaomi MI Band 2. It will get the walking activity data from Xiaomi MI Band 2 and update it into the database.

Then the Me agent includes an dialogue-based platform, Slack and pre-trained Personalized Exercise Training Chatbot. It will notify the user when a user needs to walk and get the user's physical, mood states if the user delay or cancel the exercise notification. Finally, the database part is based on Google Sheet API. It is used to save and update the walking activity data from Xiaomi MI Band 2 and states elicited from the user or inferred by the Me agent.

Chapter 2

System Design

2.1 Original Schema

A lot of different persuasion systems have been presented because of the huge demand for motivating the users to exercise. Berardina and Irene (2011) [11] introduced a persuasion system to instruct the user exercise properly and continually in a start environment in novel work that has been continued in the user modeling, adaptation and personalization (UMAP) community. The system's goal is to motivate the user to exercise more in a fitness center where contains different smart device agents by sending motivation notification in appropriate time based on the data from smart device agents. Therefore, the Me agent could real-time monitor a user's activity and send him motivation notification after getting data from sensor devices, such as heart rate, running speed and so on. Its main focus is in the real-time activity to guard and motivate the user exercise more and properly.

Inspired by the persuasion system designed by Berardina and Irene (2011) [11], we adjusted the system according to our demand of personalized exercise training chatbot by adding calendar module, wear detection module, physical states, emotional states and specific data from Xiaomi MI Band 2, such as steps, speed, heart rate and calories. Different with their goal that real-time motivate the user exercise continually and instruct the user exercise properly, we only focus on the average activity per day or per week but not real-time monitor

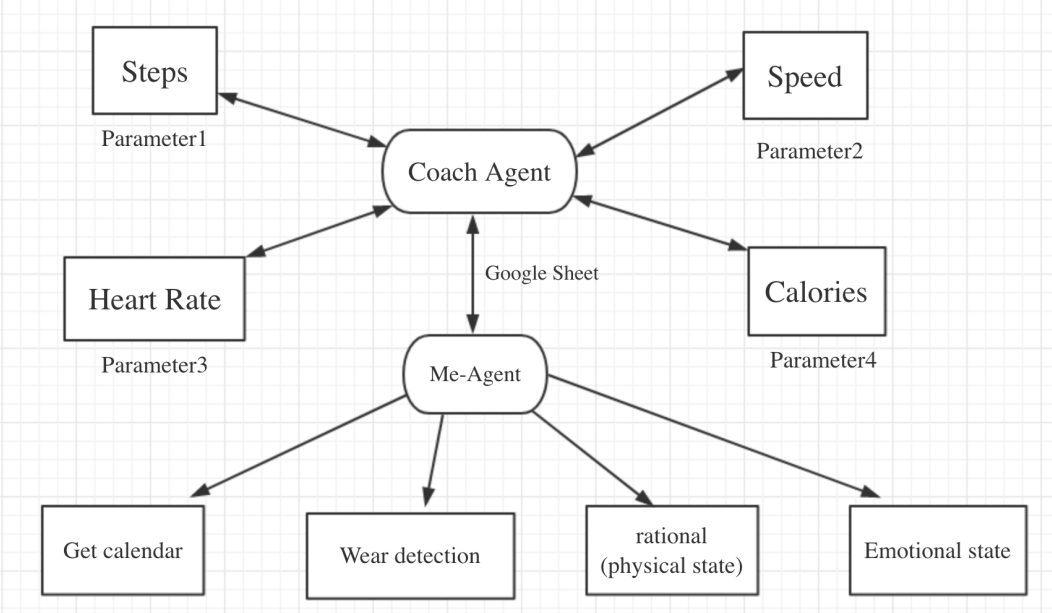

Figure 2.1: *Original system schema*

and give the chance for the user to give feedback instead of just sending the user notification or motivation message. Also, the user will not be limited only being motivated in a smart environment because we use wearable fitness devices instead of smart devices in a fitness center. Thus, we will move the sensor agents and device agents out as wearable fitness devices contain them together already. What's more, we will only care about the activity data so far and compare them with goal activity that set by the user so that we don't need real-time monitor data, such as heart rate and speed parameters from wearable fitness devices anymore. Considering the possible factors or personal staff that influence the user's exercise, we add Google Calendar module, physical status, and emotional status into the system. the Me agent needs the user's history status such as physical status to decide what kind of motivation message it should send, then we add Google Sheet as the database to save all the activity data from wearable fitness devices and status of the user.

In conclusion, the whole system includes the Coach agent, the Me agent and the database part. The Coach agent represents Android mobile application connected with Xiaomi MI Band 2 and gets the data of steps, heart rate, speed and calories. the Me agent represents the Personalized Exercise Training Chatbot gets access of Google Calendar of the user and

update the physical states and emotional states into the database. The database part is designed to save all the data collected from Xiaomi MI Band 2 and the Me agent. The original system schema is shown as Figure 2.1.

2.2 Updated Schema

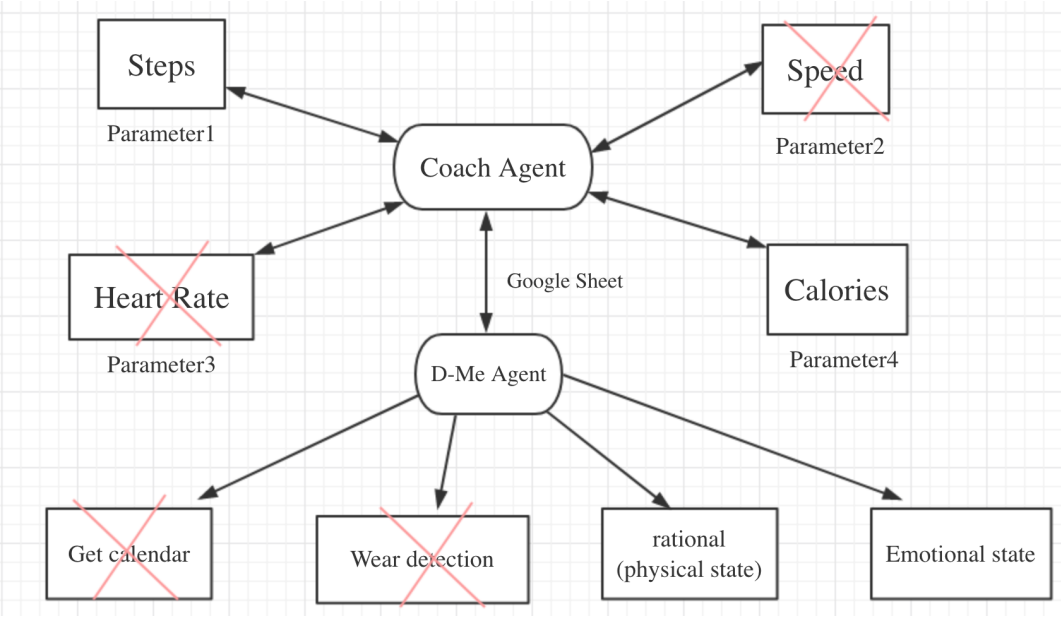

Figure 2.2: *Updated system schema*

Again, one of our goals is to motivate the user to exercise more, that is walk or run more based on Xiaomi MI Band 2 is been used. We need to care more about how many steps that the user has per day instead of the monitoring real-time status of the user's walking or running, such as speed or heart rate. Then we will move the speed and heart rate part out. Another goal of this system is ask the user to give feedback and analyze the user's feedback to generate specific motivation messages. Thus, instead of detecting wearable fitness status and reading the user's Google Calendar, the Me agent in the system will ask the user directly and get the user's answer. Therefore, we will delete get calendar module and wear detection module. The steps and calories will be kept in the updated system. Steps will be fetched by the Coach agent through Bluetooth and calories is calculated using Steps got from the

Coach agent. Then we can update steps and calories in Google Sheets, using it as an online database.

In conclusion, we will delete the part of speed, get calendar, wear the wrist band modules and put emotional and physical states together. Then the updated schema will be shown as Figure 2.2.

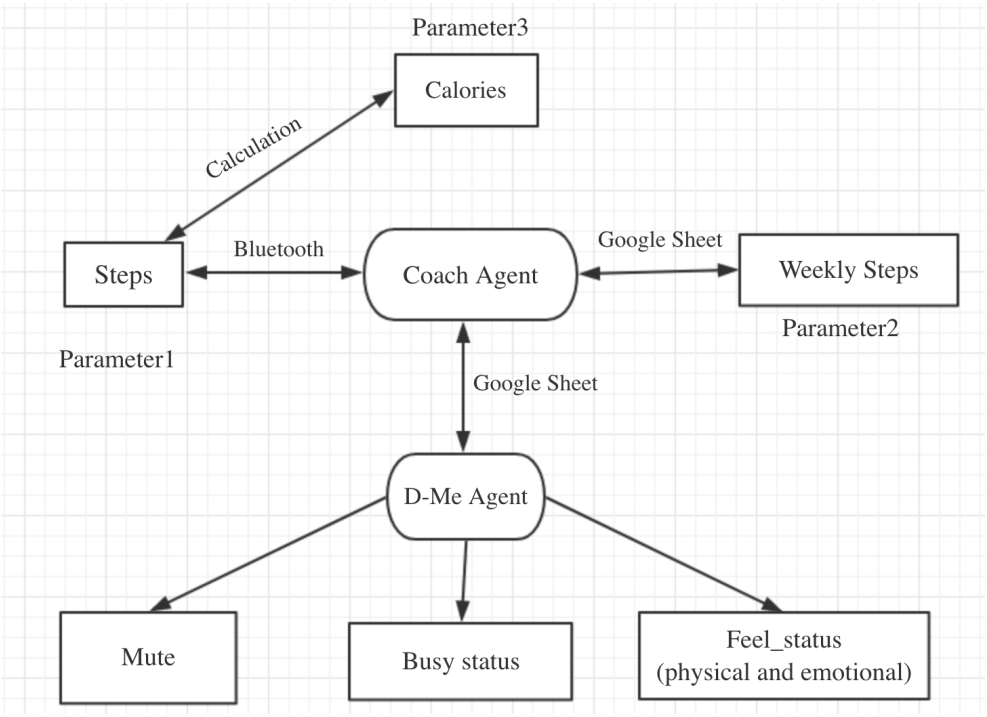

Figure 2.3: *Final system schema*

Since the motivation message will be the same for rational and emotional status, we merge these two modules and rename it as *Feel_state*. We also add mute module and busy status module for the Me agent since we can get more feedback from the user and return to influence motivation messages. Mute module allows the user to mute the Me agent when necessary such as the user need to prepare for a very important exam and so on. Busy status make the Me agent understand that the user is busy or not. If the user is in the busy state, depends on the levels of busy, the exercise notification will be delayed different related time. The Coach agent gets parameters by Bluetooth connection with wearable equipment. The final schema for Personalized Exercise Training Chatbot is shown as Figure 2.3.

Chapter 3

The Coach Agent Based on Wearable Equipment

This chapter will describe details about how the Coach agent gets connection with wearable equipment Xiaomi MI Band 2 through Bluetooth and update data from Xiaomi MI Band 2 to the database, Google Sheet. The schema about their relationship is shown as Figure 2.3 in Chapter 2.2.

3.1 Xiaomi MI Band 2

There are hundreds of smart wearable wristband equipment on the market now and the most popular wearable brands are Apple, Garmin, Fitbit and Xiaomi.[12] According to the compatibility of Android system, we only consider and compare basic wearable wristbands in Garmin, Fitbit and Xiaomi. We choose Xiaomi MI Band 2 , Garmin Vivofit and Fitbit Charge 2 to compare parameters between them. The Appearances of three wristbands[13] [14] [15] are shown as Figure 3.1. The parameter comparison is shown as Table 3.1.[16]

Based on the parameters needed for the Coach agent shown in Figure 2.3 in Chapter 2.2. All the wristbands in Table 3.1 satisfy the requirement that contain steps, weekly steps and calories. However, Xiaomi MI Band 2 is the cheapest and last much longer time than Fitbit

Figure 3.1: *Appearances of three wristbands*

Charge 2. What's more, it is rechargeable while Garmin Vivofit is not even though Garmin Vivofit lasts for longer time in one year. In conclusion, we choose Xiaomi MI Band 2 as our wearable equipment after considering the features fit, cost and chargeable ability.

Table 3.1: *Comparison of wristbands among three famous brands*

Model	Xiaomi MI Band 2	Fitbit Charge 2	Garmin Vivofit
Display Material	OLED	OLED	LCD
Display Size	0.42 inch	0.83 inch	25.5 mm x 10 mm
Charging method	USB charger base	USB charger base	No
Bluetooth Version	Bluetooth 4.0	Bluetooth 4.2	Bluetooth 4.2
Battery Capacity	70mAh	90mAh	260mAh
Battery Life	Up to 20 days	Up to 5 days	1 year +
Charging Time	3h	1-2h	No
IP67 Waterproof	Y	Y	Y
Real-time Steps	Y	Y	Y
Distance	Y	Y	Y
Heart Rate	Y	N	Y
Sleep Mode	Y	Y	Y
Calories	Y	Y	Y
Activity	N	Y	N
Wifi	N	Y	N
Price($)	26.99	149.95	52.44

3.2 Android Studio

Android Studio[17] is an official application integrated development environment (IDE) for Android development. Before Android Studio were released, naive Android Development has been developed by using Eclipse Android Development Tools (ADT).[18] With Android Studio, we don't even need a real Android device to use it because it provides an emulator that simulates a real-time device and display it on our own laptops. What's more, there are many different models of Android Mobile devices that we can choose. Further, the emulator takes advantages of available processors in computer that makes it even faster when compared to a connected hardware Android Mobile device sometimes. Therefore, we used emulator to test connection with Google Sheet API and adjust the interface. At the same time, we used a hardware Android Mobile device, Meizu Note to debug connection with Xiaomi MI Band 2 and get the data from it because the emulator in Android Studio has all functions compared to a real device except Bluetooth function which is the key about connection with Xiaomi MI Band 2.

The latest version is the Android Studio 3.0 which has Gradle-based build support and rich layout editor that allows to drag and drop the interface components directly, such as Buttons, TextView and so on. Android Studio 3.0 is been used in this publication. The system requirements for Android Studio Version 3.x is shown as Table 3.2.[18]

Table 3.2: *System requirement for Android Studio*

Operating System	Windows 7/8/10 (32-bit or 64-bit) OS X 10.10 - 10.13 GNOME or KDE desktop Linux
RAM	3 GB +
Space for Android SDK	1.5 GB+
Java version	Java Development Kit (JDK) 8
Screen requirement	At least 1280*800

3.3 Data Retrieval through Bluetooth

The most important part of the Coach agent is connecting with Xiaomi MI Band 2 and fetching data from it through Bluetooth. We will describe how to use Android Studio to connect our Meizu Note device with Xiaomi MI Band 2 through Bluetooth and then how to fetch data from Xiaomi MI Band 2.

3.3.1 Bluetooth Connection

Bluetooth is a wireless technology standard for exchanging data over short distances from fixed or mobile devices, and building personal area networks(PANs). It is mainly designed for low-power consumption, with a short range based on low-cost transceiver microchips in every device.[19] The distance range is depended on its power, however, real useful ranges vary in real-world. The speeds and distance ranges based on the Bluetooth version are shown as Table 3.3.

Table 3.3: *Ranges based on Bluetooth version*

Bluetooth version	Maximum speed	Maximum range
3.0	25 Mbit/s	10 meters
4.0	25 Mbit/s	60 meters
5	50 Mbit/s	240 meters

Android Studio gives developer access to the Bluetooth functionality of Android device through the Android Bluetooth APIs. For general Bluetooth devices pairing, we need to set up Bluetooth, discover and scan for the devices, try to pair the device neither as a server or a client. In this publication, since we have connected the Xiaomi MI Band 2 before, the process will be adjusted into setting up Bluetooth, finding the Xiaomi MI Band 2 in local paired devices list, trying to pair Xiaomi MI Band 2, setting Xiaomi MI Band 2 authentication and fetching the data from it.[20]

Setting up Bluetooth

At first, we need to give the permission of Bluetooth API to Android application in Android Studio. In order to do that, we add permission of Bluetooth, Bluetooth-Admin and Access-coarse-location in AndroidManifest.xml file which saves all the permissions needed for running the application.

Then we enable the default Bluetooth adapter of the Mobile device and call it if the device supports Bluetooth service. Then the application will pop out a notification whether active the Bluetooth function if the Bluetooth is not active yet.

After enabling Bluetooth service, we will try to find MAC address of Xiaomi MI Band 2 by looking up "MI Band 2" in names of local Bluetooth paired devices list, which is unique for each Bluetooth device. The reason we find MAC address of Xiaomi MI Band 2 directly instead of scanning Bluetooth devices surround is it is much faster on the process of connecting Xiaomi MI Band 2 since we have connected to the Xiaomi MI Band 2 before.

After getting the MAC address of Xiaomi MI Band 2, we will connect with it by using connectGatt service of bluetoothGatt. BluetoothGatt is a class that provides Bluetooth GATT functionality to enable communication with Bluetooth Smart or Smart Ready devices. When connectGatt service is been detected as active then it means that we connect Xiaomi MI Band 2. Once Xiaomi MI Band 2 is connected, we can get the data about the hardware and battery info. However, we still can not fetch the user data from Xiaomi MI Band 2, such as steps and activity data even though it is been connected with Android Mobile device. Because we need get authentication from Xiaomi MI Band 2 before it sends the user data to Mobile Android device.

3.3.2 Data Retrieval from MI Band 2

Authentication

After connecting with Xiaomi MI Band 2, we can read the basic information from it already, such as device info, device name, device version and so on. But it is not enough for our goal.

We need to read the user data, such as real-time steps. In order to achieve that, it is necessary to get authentication from Xiaomi MI Band 2 in three steps. First, sending a "secret" key to it, we will use byte 0 of the byte data. Then it will request a random authentication key from the band. Finally, it will receive the encrypted random authentication key and send it back to Xiaomi MI Band 2. If all the parts work properly, Xiaomi MI Band 2 will give the mobile device authentication access after receiving the same authentication key that it sent to Mobile device.

Data Retrieval

Table 3.4: *UUID characters of "fee0" UUID service*

UUID character	Data
00000003-0000-3512-2118-0009af100700	CONFIGURATION
00000005-0000-3512-2118-0009af100700	ACTIVITY DATA
00000006-0000-3512-2118-0009af100700	BATTERY INFO
00000007-0000-3512-2118-0009af100700	REALTIME STEPS
00000008-0000-3512-2118-0009af100700	USER SETTINGS

Table 3.5: *Each bit and meaning of battery "0006" UUID characters*

bit	meaning
data[1]	Battery percent
data[2]	Charging status
data[9]	Charging times
data[10]	Year of last charge time
data[11]	Month of last charge time
data[12]	Day of last charge time
data[13]	Hour of last charge time
data[14]	Minute of last charge time
data[15]	Second of last charge time
data[16]	Battery percent remain after last charge time

Xiaomi MI Band 2 saved most of the data into a specific universally unique identifier(UUID) service which is a 128-bit number to identify information in computer systems. One UUID service contains many UUID characters that include different informa-

tion. The data we need are all in the same service that UUID is "0000fee0-0000-1000-8000-00805f9b34fb". Important UUID characters included in that service are shown as Table 3.4.[21] Then we can use getValue() function to get data through corresponding UUID characters. However, we still need to figure out what each bite means in the byte data to format data that are useful. Details about what each bit presents in battery service are shown as Table 3.5.

3.4 Interface Design

Android Studio provides Gradle-based build support and rich layout editor that allows to drag and drop the interface components directly, such as Buttons, TextView and so on. According to the functions, we designed five buttons that control connection, getting battery info, getting real-time steps and calories, calling Google Sheet API, draw weekly steps histogram separately. The screenshot of interface is shown as Figure 3.2.

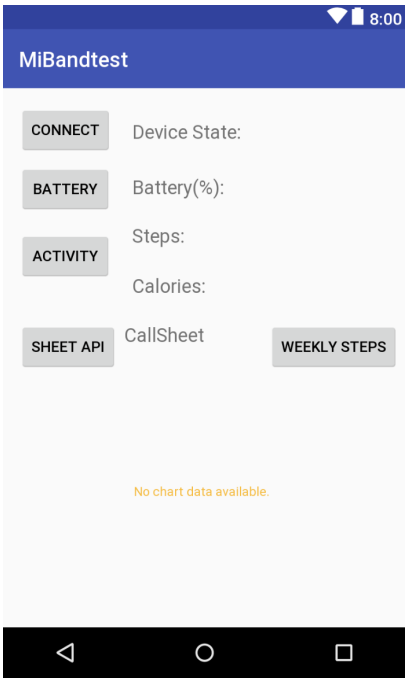

Figure 3.2: *Android application interface design*

"Connect" button calls Bluetooth connection function and "Device State" TextView will

show "connected" or "Disconnected" based on the state of Bluetooth connection. "Battery" button calls getBattery function to get percent of battery and show it in "Batteryinfo" TextView. "Activity" Button calls getSteps function to get the real-time steps and calculate calories based on the steps, then show them in "Steps" and "Calories" TextViews.[22] "SheetAPI" button calls getResultsFromApi function to update steps and battery data to Google Sheet and also get the weekly steps data from Google Sheet. "Weeklysteps" button calls startBarCharting function to draw a histogram based on the weekly data get from Google Sheet. The screenshot when the application works is shown as Figure 3.3.

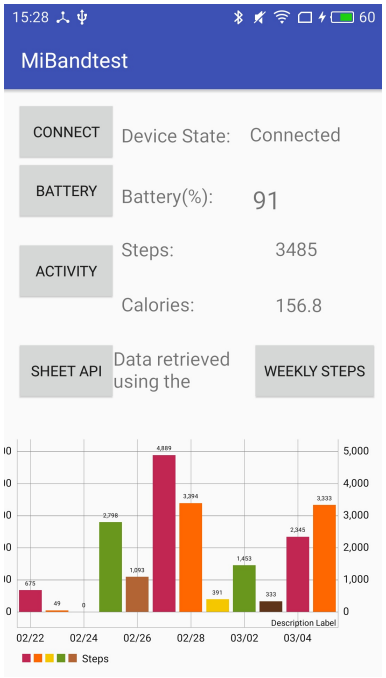

Figure 3.3: *Android application interface when working*

3.5 Android Studio and Google Sheet Connection

Similar with Bluetooth connection, it first needs to add permission of Internet, Network State and Get Accounts in AndroidManifest.xml file which saves all the permissions needed for running the application. Then we have permission to connect Internet and get the user's Google account information.

After confirming the internet connection, it will pop a notification to choose Google account if everything works fine for the first time logging in. The screenshot of choosing Google account at first time logging in is shown as Figure 3.4.

Figure 3.4: *Android application interface about account logging*

If Google service is been activated, then we can get or update data with Google sheet in three steps. First, get the Google Sheet ID from Google Sheet link. Then choose sheet number and set range of Google sheet. The pseudocode is shown as Listing 3.1.

```
// read data
spreadsheetId = "1zRYonLSPg7KqeckrHQfFfufqeVp73c6 IDGGRIPvhhA" ;
range = "Sheet2!A2:B";
data = service.get(spreadsheetId,range)
// update data
service.update(spreadsheetId,range,new data)
```

Listing 3.1: *Read or update Google Sheet - pseudocode*

.

Chapter 4

The Me Agent based on NLTK

This chapter will describe details about how the Me agent tags sentences from the user and update the information or status get by sentiment analysis into the database Google Sheet. Also, the rules create for the Me agent and how the Me agent will be activated or trigged, and communicate with the user will be introduced. The schema about their relationship is shown as Figure 2.3 in Chapter 2.2.

4.1 Slack

Slack[23] is a cloud-based set of proprietary team collaboration tools and services. It already becomes a popular internal community platform used by communities, groups or teams because it is a free platform available for an unlimited number of the users per channel. However, the users can upgrade to various paid versions to get access to bigger channels or additional features. It allows team members join through a specific URL or invitation sent by a team admin or owner. What's more, the user can communicate with his team members in public channels or private channel without the use of email or group SMS.

One other reason for choosing Slack as interactive dialog platform of the Me agent is because Slack synchronizes channel messages to IOS, Android and Web service. Thus, it has less chance for the users to miss the notification or message from the Me agent. What's

more, Slack provides API for bot users to allow bot users post real-time messages and react to the users. In the future, we could also set the bot user into app bots that will be easier to install, distribute and combine with other features of Slack platform if the system becomes more stable.

4.2 NLTK

NLTK (Natural Language Toolkit) is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries.[24]

NLTK already becomes a popular tool dealing with Natural Language Processing. It is a free, open source toolkit working for Windows, Mac OS X and Linux. In this publication, we will use the built-in function to tokenize sentences and do sentiment analysis with them. Except that, we also use libraries of NLTK for classifying and training the classifier to do sentiment analysis which will be saved in the label of status in Google Sheet. Then the Me agent will motivate the user to exercise more or close the notification based on the status we have labeled in the Google Sheet.

4.3 Communication with User

The most important part of the Me agent is to communicate with the user based on activity data from Xiaomi MI Band 2 in the database, Google Sheet. We will describe the basic rules of the Me agent, sentiment analysis with the sentences from the user, updating Google Sheet and additional rules about motivating and notifying based on the status of the user.

4.3.1 Basic Rules of the Me Agent

Rules related to *mute*

1. Check if the Me agent is been muted based on the status of "Mute" in Google Sheet. If the status of 'Mute' > 0 that means that the Me agent is been muted, do not send any message. Otherwise, continue the coming rules.

2. Check each sentence from the user's answer whether it includes "Mute". If it contains and it is not NEGATE meaning, then we will update the "Mute" status in Google Sheet into '1' value.

3. At 11:30pm every day, update the "Mute" status $= Mute - 1$ if it is not 0.

Rules related to *Goal-steps* , *Real-Time Steps* and *Status*

1. This rule about setting up goal steps per day is a start point for the Me agent. We will first check the "Goal-steps" is Null or not. If the value of 'Goal-steps' is Null, to start the process, by sending "Hi, this is your PBT, Bob. I am here to motivate you to walk more and have a healthier life style."

2. Ask about the 'Goal-steps' by sending "What is your goal steps everyday?(recommend 5000 steps for adults)" , then save the number into the cell of 'Goal-steps' in Google Sheet.

3. If the steps $\geq$ (goal steps /2) and the time is before noon. Then send a random encouraging notification from a set of half way encouraging notifications, such as "You are doing great! Half way done of your goal steps. Keep rocking like this! "

4. If the steps $\geq$ goal steps, then send a random compliment from a set of complete compliments, such as "You have reached your goal steps today. I guess a future fitness star is coming! "

5. If the time is 6pm already and the "Real-time steps" $\leq$ ("Goal-steps"/2), if the status of "Feel-status" is 0, then we will send a random care notification from a set of few-care

notifications, such as "You did not walk too much today. Still have more than half way to go. Are you feeling Okay? "

Then we will wait for the user's answer and do sentiment analysis as discussed in 4.3.2 about the answer, then update the status to "Feel-status". If it is "Positive", we will update the value of "Feel-status" into 2 and send a random motivation message from a set of positive motivation messages, such as "Great that you are feeling good! It is good to run with a perfect mood." If it is "Negative", we will update the value of "Feel-status" into -2 and send a random comfort message from a set of comfort message, such as "Sorry to hear about that, hope you will be better soon. But I will always be here to with you."

6. At 11:30pm every day, update the "Feel-status" status = $Mute + 1$ if it is -2, update the "Feel-status" status = $Mute - 1$ if it is 2.

7. At 10am, check if the "Feel-status" is 1 and the steps yesterday is higher than "Goal-steps", then send a random encouraging message in a set of encouraging message, such as "You did a great job yesterday. What's more, running will make you feel better and love this world more."

Rules not related to *status* and *steps*

1. If use typed is not in the menu.(includes" mute and change goal") Send a message "Are you looking for the menu to mute me or change goal steps?"

2. Then check if the answer of the user contains "mute" or "goal", then it will update mute status or go the rule about setting up goal steps.

4.3.2 Sentiment Analysis

According to the requirement of the rules as discussed in section 4.3.1, we need to do the sentiment analysis of the answer of the user and update the status to Google Sheet. In

this publication, we first use NLTK library SentimenIntensityAnalyzer, and then try to use build-in function and corpora in NLTK to train classify and do the sentiment analysis.

NLTK Library: SentimentIntensityAnalyzer

Nltk.sentiment.vader is an inbuilt sentiment analyzer module that can analyze a piece of text and classify the sentences under positive, negative and neutral polarity of sentiments in NLTK using NLTK features and classifiers.. VADER (Valence Aware Dictionary for Sentiment Reasoning) is a valence aware dictionary for sentiment analysis. Because it is constructed from a generalizable , valence-based, human-curated gold standard sentiment lexicon, it does not require training data and the speed is fast enough to be used online with streaming data. Also it works well on social media style text and easily generalizes to multiple domains. It gives a sentiment intensity score to sentences, especially we could get possibility for positive, negative and neutral that sum is 1 based on a dictionary of different words and weight. In order to form VADER, first, constructing a list inspired by examining existing well-established sentiment word-banks (LIWC, ANEW, and GI). Second, incorporate numerous lexical features common to sentiment expression in microblogs that include a full list of Western-style emoticons. Third, collected intensity ratings on each of lexical features from ten independent human raters. Then screen the lexical features based on condition that remain only abs(average rate) is bigger than 2.5 out of 4. After that, ask more people to Identify Generalizable Heuristics workers to assess sentiment intensity based on some dataset in tweets and identify general heuristics. In the end, add the mean differences between each distribution into VADER's rule-based model.[25]

We created a small testing dataset about the mood and health condition both to test the accuracy of this method. The result about accuracy of testing test dataset is been shown in section 6.2.1.

Training Dataset Method

Naive Bayes[26] is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods. Bayes theorem provides a way of calculating posterior probability $P(c|x)$ from $P(c)$, $P(x)$ and $P(x|c)$.

- $P(c|x) = \frac{P(c)P(x|c)}{P(x)}$
- $P(c|x)$ is the posterior probability of class (c, target) given predictor (x, attributes).
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the likelihood which is the probability of predictor given class.
- $P(x)$ is the prior probability of predictor.

Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes.[27]

Then we will use NLTK build-in classifier based on Naive Bayes classifier to train the classifier based on training dataset or corpora that we choose. Then we still use the small dataset that we created to test the accuracy of this method. The result is been shown in Section 6.2.1.

4.4 Slack and Google Sheet Connection

Similar with Android Studio and Google Sheet Connection in Section 3.5, it first makes sure that the device running the python code is connected with internet. Then we need to turn

on the Google Sheets API, install the Google client library and call the specific Google sheet that we are going to use.

Turn on the Google Sheets API

- Create or select a project in the Google Developers Console and open Google Sheets API.Created a credential and choose Web server option.
- Give the access to application data and select role as Project editor which allows the user to edit the data.
- Then it will download a JSON file in the laptop. Find value of *"client-email"* and share the Google Sheet we are going to use in the project with it.[28]

Install the Google Client Library

Go to the terminal and use commend $*pip install gspread oauth2client* to install Google Sheet client library and service.

Call the Google Sheet

First, we will import "gspread" library and oauth2client service. Then we create credentials and client for it. Finally, we have connected Google Sheet and have access to the specific Google sheet that we are going to use. The Python code is shown as Listing 4.1.

```
import gspread, oauth2client, ServiceAccountCredentials
scope = ['https://spreadsheets.google.com/feeds']
creds = ServiceAccountCredentials.('PBTconct.json', scope)
client = gspread.authorize(creds)
sheet = client.open('PBT_user').sheet1
PBTInfo = sheet.get_all_records()
```

Listing 4.1: *Call the specific Google Sheet - Python code*

Chapter 5

Implementation

This chapter will descript how each part of the system is been implemented, what problem we have met and how to solve them. In the end, we will run the whole system and record the results.

5.1 Connection between MI Band 2 with Meizu Note

We used Android Studio to compile mobile Android application into Meizu Note. Since we need to detect whether the Bluetooth is open and would send a notification to open Bluetooth if it is not open. However, the buttons will not work if open Bluetooth after opening the application. The reason is that when we keep checking whether the Bluetooth service is working or not, if not will send the notification. But it will not get out of the loop even after we say yes and open Bluetooth already. Therefore, we need to confirm that Bluetooth is open before we open the application. Otherwise it will not work well or interrupt after we click the button.

After confirmation with Bluetooth function is opened, we would try to connect with the MI Band 2. It is fast to connect since we don't search by scanning Bluetooth devices but get the Mac address directly through paired devices list of Meizu Note. Then we try to read the battery data and the data we get contains other bite information, such as how long it

has been charged last time and how many times it has been charged. But it is not necessary, then we just pick data[1] as our battery data. The results before we formatted battery data and after are shown as Figure 5.1 and Figure 3.3.

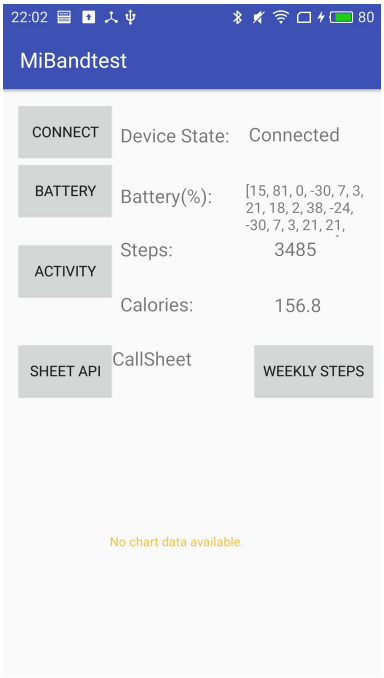

Figure 5.1: *Battery data before we format*

After connecting with MI Band 2 and get the battery data, I thought the other data like steps should be easy to fetch since we just need to know the UUID for each activity data that we need. However, we can not get the steps and other activity data because those data belongs to the user data which needs authentication of MI Band 2. Then we have to follow the protocol of MI Band 2 and get authentication to fetch steps. We will use calories = steps * 0.045 to calculate calories. Then we will update the data of activity and battery to Google Sheet. Next, we will reading the weekly data from Google Sheet and draw histogram in Mobile Application in Meizu Note device. Finally, the steps accuracy of MI Band 2 and the delay for updating and reading the data with Google sheet will be discussed in Section 6.1.

5.2 The Chatbot in Slack

We first export the chatbot parameters into the system, such as chatbot token, name and ID. Then we will run the python code to motivate the chatbot. The chatbot will be online after running the python code if everything works fine. It can say Hi and Goodbye with user. When it is the first time for the user to start using the whole system, the chatbot will say hi and introduce itself, then ask about user's goal steps. Once it detects integer, will tell the user the new goal steps and update it to Google Sheet. The screenshot of dialogue about setting up goal steps is shown as Figure 5.2.

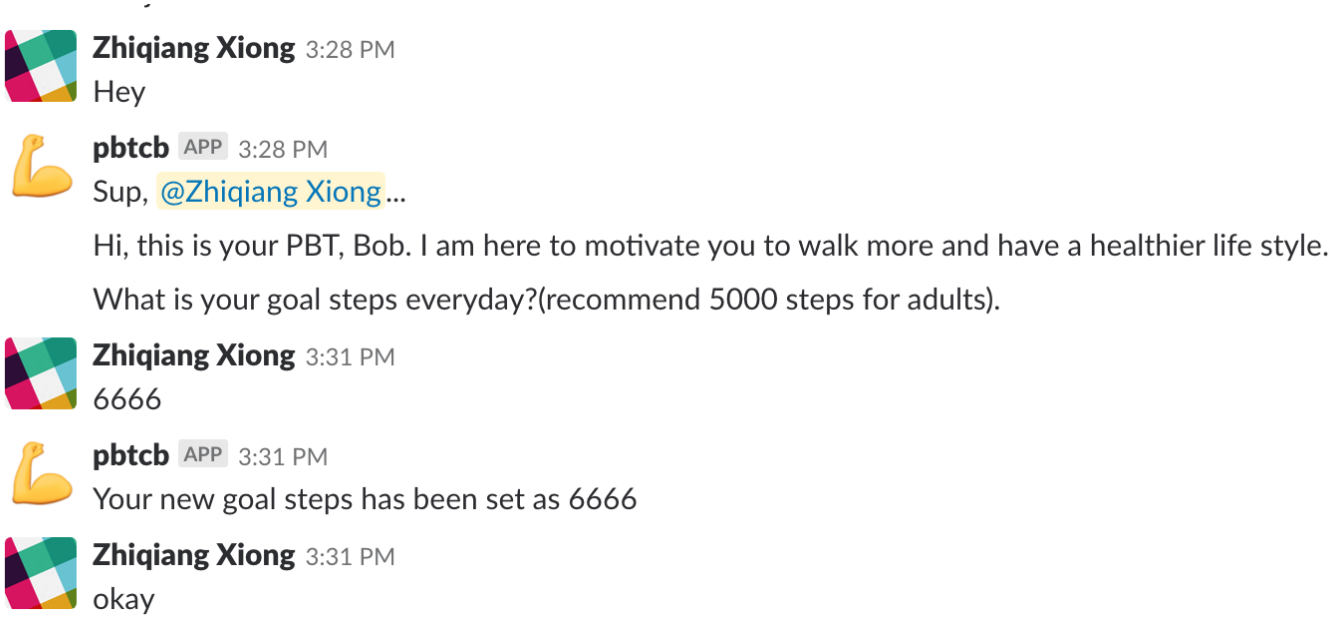

Figure 5.2: *Dialogue about setting up goal steps*

When the preconditions of a notification rule been satisfied, the chatbot will send me the message about motivation or encouragement.[29] Then we can talk with the chatbot if it was asking me based on the steps. However, we found it will not understand the answer from the user sometimes. The screenshot of dialogue about answering the motivation message is shown as Figure 5.3.

Therefore, the chatbot works well in the whole system and was motivated by the rules. However, we also found out there were many times that chatbot maybe not understand

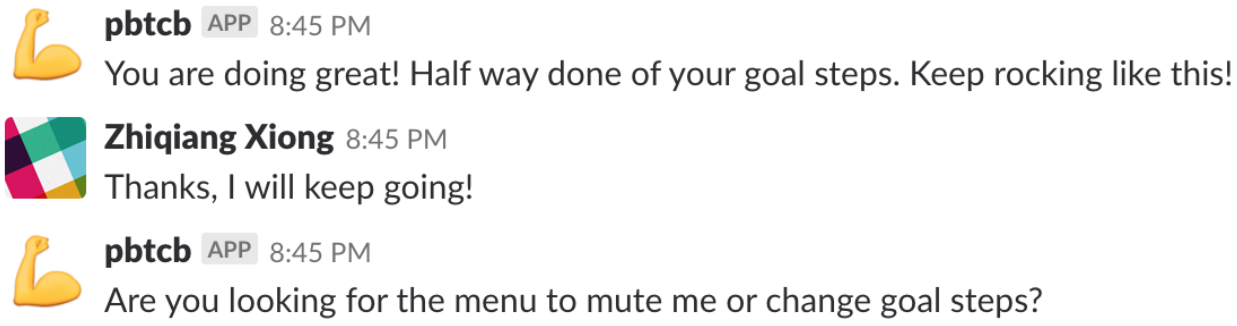

Figure 5.3: *Dialogue about not in the menu*

what user was saying if the answer is kind of complicated. What's more, the analysis about performance of the chatbot, sentiment analysis and delay of exchange data with Google Sheet is been discussed in Section 6.

Chapter 6

Testing and Result Validation of System

This chapter will first describe testing experiments includes accuracy of MI Band 2, delay for exchanging data between Google Sheet with Mobile Application of Meizu Note or Slack. Then we compare sentiment analysis results between SentimentIntensityAnalyzer library in NLTK with Naive Bayes training algorithm. Finally, we compare some dialogues results between the user and chatbot.

6.1 Testing

6.1.1 Accuracy of MI Band 2

We tried to walk 1000 steps and recorded the steps in MI Band 2 every 100 steps. The result is shown as the Table 6.1. We can see the average accuracy of Xiaomi MI Band 2 is 95.0% .

6.1.2 Delay about Exchange Data with Google Sheet

We made experiments that record the time when values were updated in Google sheet after click the buttons in Mobile application to test delay about updating values in Google Sheet.

Table 6.1: *Steps accuracy of MI Band 2*

Actual steps	Measured steps	Accuracy percent
100	105	95.0%
200	203	98.5%
300	302	99.3%
400	384	96.0%
500	488	97.6%
600	584	97.3%
700	665	95.0%
800	765	95.6%
900	851	94.5%
1000	950	95.0%

Another similar experiments are reading values from Google Sheet into Mobile application and combine them both. The result is shown as the Table 6.2. We can see the average delay of Mobile Android Application for writing or reading with Google Sheet is less than 1s and the delay is around 1s even write and read at the same time. Therefore, delay of Android Mobile application for exchanging data with Google Sheet is acceptable.

Table 6.2: *Delay of exchanging data with Google Sheet*

Delay(s)	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	Average
Android write	1.15	1.29	1.10	0.66	0.68	0.75	0.82	0.73	0.63	0.64	0.845
Android read	1.06	0.63	0.60	1.28	0.62	0.62	0.75	0.75	0.56	0.52	0.739
Android write & read	1.40	0.95	1.40	0.86	1.10	1.42	0.83	1.02	0.79	0.90	1.067
Slack write	2.52	1.75	1.73	2.03	1.76	2.80	1.55	2.63	1.99	2.06	2.082
Slack read	2.98	2.19	1.65	1.92	3.08	1.72	1.88	1.65	2.16	1.77	2.1
Slack write & read	3.15	2.43	2.02	2.73	1.96	2.85	3.05	2.13	2.86	2.03	2.521

Similarly, we made experiments between Slack platform with Google Sheet and the result is shown as Table 6.2. We can see the average delay of Slack for writing or reading with Google Sheet is around 2s and the delay is around 2.5s when write and read at the same time. The delay of Slack platform is a little higher than that of Android Mobile application, but it is still acceptable because we don't have strict limit for real-time monitoring.

6.2 Compare Results

6.2.1 Compare SentimentIntensityAnalyzer with Naive Bayes

We created a small dataset that contains 50 positive and 50 negative sentences about emotional and also physical status to test the accuracy of SentimentIntensity Analyzer and Naive Bayes algorithm. The reason we choose 100 sentences in total is that we don't focus on sentiment analysis in different users but for a specific user's talking style and how each method works. What's more, in future research, it is necessary to test bigger dataset and see how they works if we would like to apply the system in multiple users.

SentimentIntensityAnalyzer

```
I just got an interview, I am so excited today!
compound: 0.528 neg: 0.0 neu: 0.673 pos: 0.327
I'm really glad!  we're having pizza tonight!
compound: 0.5963 neg: 0.0 neu: 0.607 pos: 0.393
I'm so relieved because I did really well on my real time system exam!
compound: 0.6786 neg: 0.0 neu: 0.664 pos: 0.336
Not so good. I'm stressed about work.
compound: -0.647 neg: 0.514 neu: 0.486 pos: 0.0
Not good.
compound: -0.3412 neg: 0.706 neu: 0.294 pos: 0.0
I am sick.
compound: -0.5106 neg: 0.767 neu: 0.233 pos: 0.0
my cat sick ,I feel really sad.
compound: -0.7713 neg: 0.572 neu: 0.428 pos: 0.0
```

Figure 6.1: *Part of testing results of SentimentIntensityAnalyzer*

We give the small dataset contains 50 positive and 50 negative sentences about emotional and physical status to test SentimentIntensityAnalyzer method. The result is kind of promising actually, its accuracy is 90% based on this small dataset that contains basic general answers of the user about their emotional or physical status. The part of testing results is shown as Figure 6.1. It prints out part of the sentences following four parameters containing compound, neg, neu, pos. Compound means the intensity of the sentence and

the abs (value of compound) is higher when the intensity is higher. It will be positive number when the sentence is tagged as positive sentence, negative number when it is tagged as negative number and 0 when the sentence is neutral. Neg, neu and pos present possibility of negative sentence, neutral sentence and negative sentence separately. We will finally consider a sentence is positive if value of pos > value of neg, negative if value of pos < value of neg, otherwise it is neutral.

Naive Bayes

```
I just got an interview, I am so excited today! : pos
I'm really glad!  we're having pizza tonight! : neg
I'm so relieved because I did really well on my real time system exam! : neg
Not so good. I'm stressed about work. : neg
Not good. : pos
I am sick. : neg
my cat sick ,I feel really sad. : neg
I dont feel good, because I did not sleep well last night. : pos
```

Figure 6.2: *Part of testing results of Naive Bayes*

We used 1000 positive sentences and 1000 negative sentences in movie review corpora in NLTK to train the Naive Bayes Classifier. Then similar with the testing experiments for SentimentIntensityAnalyzer in NLTK, we give the a dataset contains 50 positive and 50 negative sentences about emotional and physical status to test Naive Bayes algorithm method. The result is kind of low, its accuracy is 60% based on this small dataset that contains basic general answers of the user about their emotional or physical status. The part of testing results is shown as Figure 6.2. The results after each testing sentence means positive tag if it is 'pos', otherwise it means negative if it is 'neg'. The results shows that when the sentences contains the negative words such as 'not', it should tag in opposite. However, the classifier did not analyze the negative words. Thus, comparing to the 90% accuracy of SentimentIntensityAnalyzer, we definitely choose the SentimentIntensityAnalyzer in this publication.

6.2.2 Dialogues between User and Chatbot

Based on the rules we created in Section 4.3.1, the Me agent will notify the user when the rules are satisfied. Such as motivation message when the user finish half of his goal or beyond it. Because that is like motivation notification without asking questions, so we don't expect that the user will answer the message. However, we noticed that maybe the user will answer the encouraging message and the Me agent will answer like "Are you looking for the menu to mute me or change goal steps?" The screenshot of dialogue about answering the motivation message is shown as Figure 6.3 Even though it is better than no answer, it is still possible to decrease possibility for the user to answer other notification messages in the future. Therefore, we need to add more rules in the future work to make the Me agent to answer more possible sentences the user might write after the motivation message.

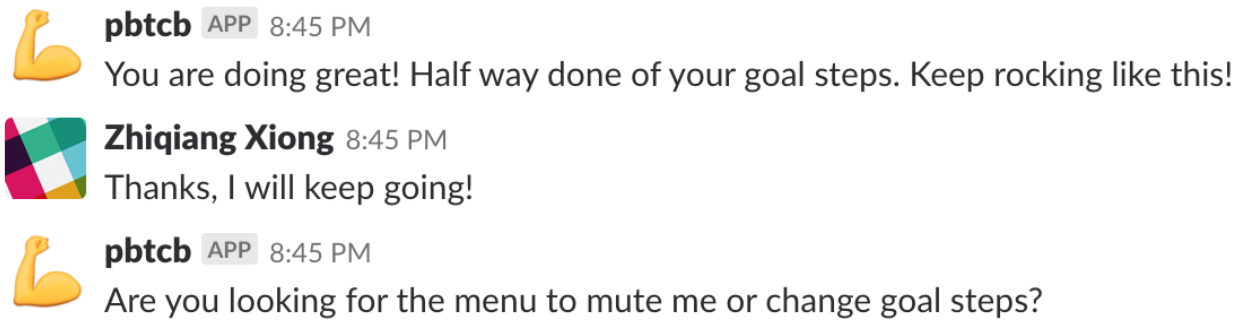

Figure 6.3: *Dialogues not in the menu*

After we compared the results of two different sentiment analysis methods, we chose SentimentIntensityAnalyzer of vader module in NLTK. According to the high accuracy of SentimentIntensityAnalyzer method, the feedback from the user could be tagged correctly so that the Me agent could send related motivating message or comforting message to the user.

The screenshot of dialogue about positive answering the motivation message is shown as Figure 6.4.We can see when we tried to ask the whether the user is okay because we found the steps were kind of low than usually. If the user's answer is positive based on sentiment analysis, we will encourage the user to run directly since he is feeling good or at least not

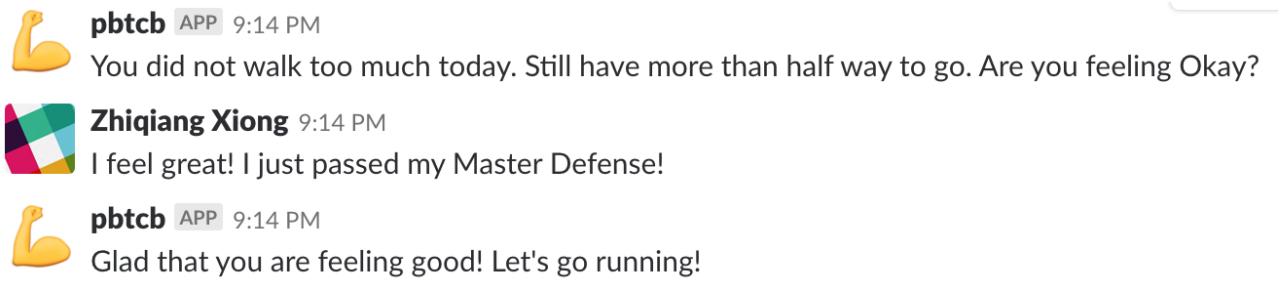

Figure 6.4: *Positive answer from the user*

bad. The screenshot of dialogue about negative answering the motivation message is shown as Figure 6.5. When the user is feeling bad, instead of motivating the user to exercise or running, we comfort the user and hope he could be better soon. Actually, after we talked with the Me agent for several times, we found maybe there are neutral status of the user, then we can motivate the user by saying "Exercise will make you happy. Maybe we should take a breath of fresh air?" Then we could add neural status besides positive and negative status in the future work to have more branches of rules.

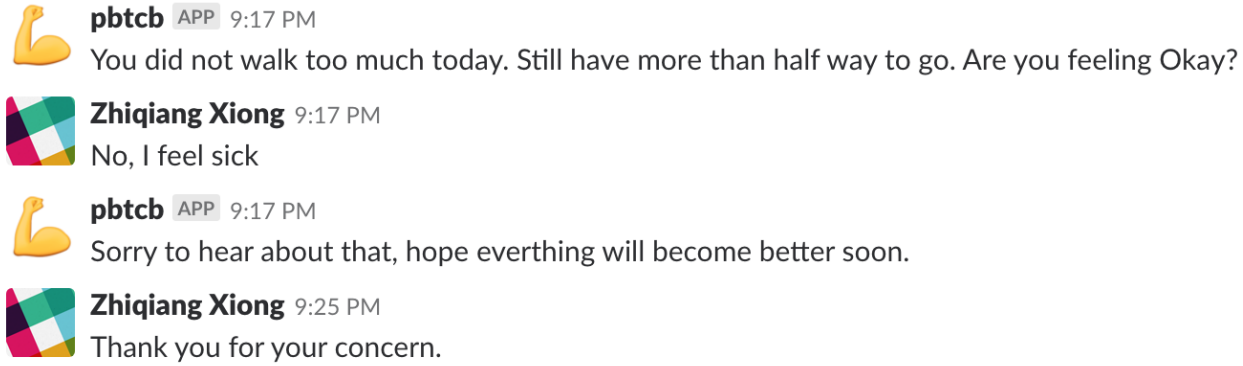

Figure 6.5: *Negative answer from the user*

Chapter 7

Summary and Future Research

In this report we have presented a personalized exercise training chatbot for individual users based on data collected from the Xiaomi MI Band 2 through Bluetooth. The whole prototype system consists Mobile application that connects with Xiaomi MI Band 2 to fetch the user's activity data and a chatbot in Slack platform to motivate the user and get feedback from the user. The Mobile application in Meizu Note device has successfully fetched the steps and battery from Xiaomi MI Band 2 and updated them into Google Sheet. On the other side, The chatbot could send the user notification and motivation messages to motivate the user, also tag the status in Google sheet by sentiment analysis on sentences from the user. It satisfied with our goal of motivating the users to exercise more by discussing exercise statistics with the user and send the user some notifications when their daily steps have satisfied some rules that we created. Then we have described the testing results and compare sentiment analysis accuracy between different Machine Learning methods.

And we still have a lot that we can do in the future research on the system. As introduced in Chapter 6.1.1, we can see the accuracy of the steps of the MI Band 2 is 95%, so we can do more experiments to get the average accuracy of MI Band 2 and adjust the results after reading data from MI Band 2. What's more, we can improve accuracy of Sentiment Analysis by training the dataset about health or mood but not just movie reviews. Finally, more other Machine Learning algorithm can be tested to decide which one fits the goal of this publication

the most.

The system will further be modifier to fetch more data from fitness device and connect with more fitness devices to get more data from the user. Therefore, we could change Xiaomi MI Band 2 into other wristband containing more features and activities.

The dialogues between the user and chatbot in Chapter 6.2.2 show that many times the chatbot could not understand the user's answer. If this kind of situation happens many times, the user will be lazy to answer every notification that leading to losing the most important feature about getting feedback from the user. Therefore, what we can do to improve is adding more rules to deal with more answers from the user. And there is a better solution that using dataset between the users and real personal body trainers to pre-train the chatbot so that it can answer most of questions about body training. But under this situation, we need to consider that the length of the conversation. Because we will have information enough after getting few answers from the user. Once we get all status that we need or the length of the conversation is long enough, then we will cut the conversation and say something like "Okay. I think you need to go running now!"

We can add another function about delay the notification based on the busy status of the user. For example, the user is working on his homework when the chatbot tries to send him a motivation message about running. The user will tell the chatbot that he is busy with his staff for right now, maybe finish it in half an hour. Then the chatbot will delay the same notification again after half an hour.

To motivate the user exercise more, society comparison will be imported into the system. The user could share his activity to his family members or friends. Then we need to add more rules about comparison with friends. The flowing are some example rules. For example, If steps belongs to [(friend's steps)-100, (friend's steps)], "Doing great, so much close to your friend, let us beat that one, should we?" will be sent. Thus, the prototype motivate system in this publication is successfully and continuously building an active platform to motivate the user exercise by Artificial Intelligence chatbot.

Bibliography

[1] Sunny Consolvo and Predrag Klasnja. Goal-setting considerations for persuasive technologies that encourage physical activity. *Persuasive Technology, Fourth International Conference 2009*, pages 1–8, 2009.

[2] Meethu Malu and Leah Findlater. Toward accessible health and fitness tracking for people with mobility impairments. *10th EAI International Conference on Pervasive Computing Technologies for Healthcare, IEEE*, page 1, 2016.

[3] Christopher C. Tsai and Kevin Patrick. Usability and feasibility of pmeb: A mobile phone application for monitoring real time caloric balance. *Mobile Networks and Applications.*, 12:2–3, 2007.

[4] Sean A. Munson and Sunny Consolvo. Exploringgoal-setting, rewards, self-monitoring, and sharing to motivate physical activity. *Pervasive Health Conference*, pages 25–32, 2012.

[5] Mark Matthews and Tanzeem Choudhury. Tracking mental well-being: Balancing rich sensing and patient needs. *Computer*, 4:36–43, 2014.

[6] Sunny Consolvo and Katherine Everitt. Design requirements for technologies that encourage physical activity. *Human-computer Interaction International Conference*, pages 457–466, 2006.

[7] Jeffery W. Robert and Wing R. Rena. Use of personal trainers and financial incentives to increase exercise in a behavioral weight-loss program. *Journal of Consulting and Clinical Psychology*, 66:777–783, 1998.

[8] Andreas Mller and Luis Roalter. A personal trainer for physical exercises. *International Conference on Pervasive Computing and Communications Workshops*, pages 1–2, 2012.

[9] Judith Masthoff and Julita Vassileva. Tutorial on personalization for behaviour change. *ACM Intelligent User Interfaces 2015 Conference*, page 439, 2015.

[10] Fabio Buttussi and Luca Chittaro. Mopet: A context-aware and user-adaptive wearable system for fitness training. *Artificial Intelligence in Medicine (AIME '07)*, pages 153–163, Feb, 2008.

[11] Berardina De Carolis and Irene Mazzotta. Motivating people in smart environments. *Proceedings of the 24th ACM Conference on User Modeling, Adaptation and Personalisation*, pages 368–381, 2011.

[12] Mike Murphy. The most popular wearables brand in the world isn't apple or fitbit. Archived at https://qz.com/1066920/xiaomi-beats-out-apple-aapl-and-fitbit-fit-as-the-most-popular-wearables-brand-in-the-world/, August 31, 2017.

[13] Xiaomi. Xiaomi mi band 2 outlook. Archived at http://www.mi.com/en/miband2/, March, 2018.

[14] Fitbit. Fitbit charge 2 outlook. Archived at https://www.fitbit.com/shop/charge2, March, 2018.

[15] Garmin. Garmin vivofit outlook. Archived at https://buy.garmin.com/en-US/US/p/143405, March, 2018.

[16] Gadgets Now (2018). Compare fitbit charge 2 vs xiaomi mi band 2 vs garmin vivofit. Archived at : https://developer.android.com/guide/topics/connectivity/bluetooth.html, April 17, 2018.

[17] Android Studio. Android studio: The official ide for android. Archived at https://developer.android.com/studio/index.html, March 11, 2018.

[18] Android Studio. Archived from the Wikipedia at https://en.wikipedia.org/wiki/Android_Studio, March 03, 2018.

[19] How bluetooth technology works. The Bluetooth Special Interest Group, January 2008. Retrieved at February, 2008.

[20] Android. Bluetooth[web log guide]. Archived at https://developer.android.com/guide/topics/connectivity/bluetooth.html, 2017.

[21] Jin. Xiao mi band protocol analyze. Archived at http://changy-.github.io/articles/xiao-mi-band-protocol-analyze.html, December 28, 2015.

[22] Timothy Banas. How to convert pedometer steps to calories. Archived at https://www.livestrong.com/article/238020-how-to-convert-pedometer-steps-to-calories/, September 11, 2017.

[23] Lee Gomes. A service built for the era of mobile phones and short text messages is changing the workplace. Archived at https://www.technologyreview.com/s/600771/10-breakthrough-technologies-2016-slack/, April 5, 2017.

[24] Steven Bird. Nltk 3.2.5 documentation. Archived at http://www.nltk.org/, September 24, 2017.

[25] C.J. Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*, pages 1–5, 2014.

[26] Naive Bayes classifier. Archived from the Wikipedia at https://en.wikipedia.org/wiki/Naive_Bayes_classifier, YEAR = Febrary 03, 2018 .

[27] Sunil Ray. 6 easy steps to learn naive bayes algorithm (with codes in python and r). Archived at https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/, September 11, 2017.

[28] Twilio (Producer). Google sheets and python. Podcast archived from https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/, Febrary 27, 2017.

[29] Fabio Buttussi and Francesco Ricci. Guiding patients in the hospital. *Proceedings of the The 19st Conference on User Modeling, Adaptation and Personalization (UMAP 2011)*, pages 309–319, 2012.