Development of a Cycling Safety Services System and Its Deep Learning Bicycle Crash Model

I-Hsuan Peng, Pei-Chun Lee, Chen-Kang Tien, and Jyun-Sen Tong

***Abstract*—This research developed an Internet of things (IoT) services system for cyclists to keep them safe when cycling – from terminal to back end – called the *cycling safety services system*. The proposed system consists of wearable devices, a service mobile app, and back-end services, primarily providing three categories of services: (1) The cycling team services, (2) the physiological status services, and (3) the environmental information service, incorporating the technologies of deep learning, IoT end device development, mobile app programming, RESTful API implementation, open data exploitation, etc. The proposed system aims at protecting the cyclists from being left out when cycling as a team, warning them when their physiological status is going to be abnormal or when there is a possibility of a bicycle crash, as well as proactively sending out urgent messages with location information when a crash occurs. Moreover, to enable the mobile app to recognize crash events, this research trained a deep learning bicycle crash model of 87.8049% accuracy and implemented a procedure in the mobile app based on this model to detect bicycle crashes automatically. The proposed system also provides a way for the cyclists to report any false crash alarm so that the crash model can be re-trained to reduce its false alarm ratio after the system is distributed to the consumers in the future. This research confirmed that the proposed system works well, and suggests that the proposed system, the crash model, the data collection method with its associated mobile apps, and the anonymous crash locations collected in the future can be valuable and contribute to the cycling society and relevant researchers in a positive way.**

***Index Terms*—AI, artificial intelligence, bicycle crash, cycling, cycling team, deep learning, Internet of things, IoT, off the team, physiological information.**

Manuscript received June 23, 2021; revised November 1, 2021; approved for publication by Jeongyeup Peak, Division III Editor, December 22, 2021.

This research was funded by the Ministry of Science and Technology, Taiwan (R.O.C.), grant numbers MOST 105-2622-E-159-004-CC3, MOST 106-2622-E-159-006-CC3, MOST 107-2622-E-159-004-CC3, MOST 108-2622-E-159-002-CC3, MOST 109-2635-E-159-002, and MOST 109-2622-E-159-004 as well as by the PAIX Design Co. Ltd.

I-H. Peng is with the Department of Multimedia and Game Development, Minghsin University of Science and Technology, Taiwan. email: ihpeng@must.edu.tw. Mailing address: No.1, Xinxing Rd., Xinfeng Hsinchu 30401, Taiwan(R.O.C). Tel.: +886-3-559-3142 ext. 1518.

P.-C. Lee, the corresponding author, is with the Department of Information Management, Minghsin University of Science and Technology, Taiwan. email: pjlee@must.edu.tw. Mailing address: No.1, Xinxing Rd., Xinfeng Hsinchu 30401, Taiwan(R.O.C). Tel.: +886-3-559-3142 ext. 3436.

C.-K. Tien was with the Department of Computer Science and Information Engineering, Minghsin University of Science and Technology, Taiwan, and is with the graduate school of Communication Engineering, National Central University, Taiwan. email: a22912907@gmail.com. Mailing address: No.1, Xinxing Rd., Xinfeng Hsinchu 30401, Taiwan(R.O.C). Tel.: +886-3-559-3142 ext. 1518.

J.-S. Tong was with the Department of Computer Science and Information Engineering, Minghsin University of Science and Technology, Taiwan, and is with the graduate school of Information and Computer Engineering, Chung Yuan Christian University, Taiwan. email: nwk.class@gmail.com. Mailing address: No.1, Xinxing Rd., Xinfeng Hsinchu 30401, Taiwan(R.O.C). Tel.: +886-3-559-3142 ext. 1518.

Digital Object Identifier: 10.23919/JCN.2022.000007

I. INTRODUCTION

IT has been a while that riding a bicycle or cycling becomes a popular activity worldwide, with purposes such as recreation, commuting, exercising, and competition. Though cycling might benefit the environment as well as human beings for it produces less pollution and – as a form of exercising – can be good for a person's health [1]–[3], it was found in newspapers, blogs, research articles in addition to the experiences of the authors' friends and relatives that some health or life threats hide behind cycling that could harm or kill a cyclist. For instance, cyclists could be uncomfortable, injured, or struck down because they continue cycling even *when their physiological status has already been abnormal and unsuitable to exercise* – such as, exceeding a certain heart rate [4], hypohydration [5], and low oxygen saturation [6] – especially when they already have cardiovascular diseases [7]. Some cases also point out that *bad air quality* could badly affect a cyclist's health [8]. Moreover, a cyclist might get seriously harmed or even killed due to a *severe crash* [9]–[18].

In this context, intensive multidisciplinary theoretical and practical research has been conducted especially concerning cycling crashes or near misses, which have apparently drawn great attention from governments, organizations, and researchers worldwide [9]–[29]. Even a standardized technical specification for the intelligent transport system (ITS) – ETSI TS 103 300-2 V2.1.1 [19] – has been officially defined by ETSI for so-called vulnerable road users, including cyclists, of course. Besides various facets of analyses and investigations performed on cycling accidents based on diverse data sources around the world as well as solutions or suggestions regarding road safety policies, cycling infrastructures, and cyclist's personal cycling safety equipment, emerging advanced computer science technologies also arouse interest of researchers and enterprises to develop technological hardware/software services to keep cyclists secure while also carrying out real-life situational data collection for further study and iterative service enhancement. For example, Ibrahim *et al.* [20], [21] proposed a computer vision-based deep learning model to detect cycling near misses. Tabei *et al.* [22] designed a system to detect cycling accidents using a support vector machines (SVM)-based algorithm with the collected magnetic, angular rate, and gravity (MARG) sensory data. The Amazon smartcycle project [24] developed a highly integrated system incorporating the technologies of the IoT, computer vision, cloud and edge computing, machine learning, etc. to detect road hazards and localize obstacles during the ride. Williams [25] innovated a mobile app that exploits the internal sensors of an iOS smartphone to detect a bicycle crash using a threshold-based detection algorithm with the assistance of Apple's CMMotionManager sensor fusion algorithms, and when a crash is recognized, automatically invoke the virtual voice assistant Amazon Alexa to guide the cyclist through the crash reporting procedure. Aside from the tech-aided facilities applied to the cyclist's side, there are also solutions designed for other vehicles on the road or the ITS, such as a vehicle open door safety system proposed by Zhu *et al.* [26], an automatic traffic monitoring and management system for pedestrians and cyclists developed by Pourhomayoun [27], and a cyclist orientation detection algorithm presented by Garcia-Venegas *et al.* [28], most of which exploit the technologies of computer vision and machine learning to identify cyclists or their moving statuses. In [29], Alvi *et al.* provided us with a thorough survey on varied IoT-based traffic accident detection systems – though not specifically focusing on cycling – using different kinds of sensors along with various sorts of detection algorithms, among which the machine learning technology is seemingly the most dominant tool used by researchers over the past decade.

Apart from all the respectable efforts made by multidisciplinary researchers around the world, this preliminary research aims at developing an integrated solution for *cycling team* safety not only tackling the bicycle crash problem but also taking care of the cyclist's physiological statuses in addition to informing the cyclist about the environmental conditions. To the best of the knowledge of the authors, there were *not* yet such an integrated solution – incorporating *multiple functional sets* such as cycling team application, cycling safety services, physiological statuses informing and alerting as well as environmental conditions reporting – being proposed in the literature. To divide and conquer the problems, this research narrowed the scope down to provide a practical and feasible solution to *recreational, amateur road biking with a team while only single-bicycle crashes* [16]–[18] *are under consideration*, where the data collected to train the bicycle crash detection model were acquired from a series of near-real experiments this research designed and conducted. To be specific, this research designed and implemented a *cycling safety services system* – from terminal to back end – to resolve certain cycling threats by offering services such as pre-crash warning, off-the-team warning, crash detection and broadcast, abnormal physiological statues warning, and bad environmental conditions warning. The proposed system is composed primarily of (1) *wearable devices*, each of which mainly comprises a microcontroller, several physiological information sensors, and a Bluetooth low energy (BLE) [30] module for transmitting the physiological information to the user's smartphone; (2) a *service mobile app* exploiting the built-in three-axis accelerometer, GPS module, and wireless transceiver modules of the user's smartphone to provide three categories of services: (a) The client side of the *cycling team services*, (b) the *physiological status services*, and (c) the *environmental information service* – taking advantage of the Taiwan governmental open data platform [31]; (3) *back-end services*, which operate as the server side of the cycling team services and are responsible for all the back-end support for the whole services. It is noteworthy that to enable the service mobile app to automatically evaluate whether a crash occurs, this research trained a deep learning [32] bicycle crash model – of 87.8049% accuracy – in advance and implemented the crash detection procedure based on this model in the mobile app. After the proposed system is commercialized and distributed to the consumers in the future, by continually collecting the false alarm feedback from users, the crash model can continue to be re-trained to decrease its false alarm ratio (FAR). Besides, the future collection and announcement of anonymous crash locations might be helpful to the cycling community because those locations can be potentially dangerous for many other cyclists, where "anonymous" means that no personal information will be recorded along with those crash locations.

The rest of this paper is organized as follows. Section II introduces system development-relevant preliminaries. Section III explains how this research designed the proposed system. Section IV describes the implementation of the proposed system. Section V elaborates on how this research developed the bicycle crash model. Finally, Section VI provides concluding remarks.

II. Preliminaries

A. System Development-Related General Knowledge for Exercising and Design Decisions

Concerning the physiological statuses, it is well known that when exercising, it is proper to maintain the target heart rate (THR) or target heart beats per minute (bpm) between a certain percentage range of the maximal heart rate (HR_{max}) as suggested in [33]. Moreover, oxygen saturation must also be kept at a certain level – 95% and above at sea level, and above 92% at 1,600 meter's altitude. If the oxygen saturation is too low, it can be harmful or even fatal [6]. Therefore, this research chose the MAX30100 sensor solution [34] – which integrates the heart rate monitor and the pulse oximetry – to detect the heart rate and oxygen saturation of a user. Furthermore, the lack of water (hypohydration or dehydration) can also cause discomfort or even critical destructive results to cyclists [5], [35]. One of the hints that a cyclist is lacking water is a high body temperature (could incur hyperthermia when getting worse). Hence, this research chose the MLX90615 infrared thermometer [36] to detect the user's body temperature. This research incorporated these three kinds of physiological information sensors with a BLE-enhanced microcontroller board to form the wearable device so as to automatically report the cyclist's physiological statuses to the mobile app for the sake of information display and abnormality warning. What is more, bicycle crashes have also been one of the most disastrous events when cycling [9]–[18]. The result can range from fracture to paralysis or even death. Consequently, this research designed and implemented pre-crash warning and crash detecting/reporting (if unfortunately occurs) mechanisms mainly by taking advantage of the mount-in three-axis accelerometer of the smartphone. Additionally, bad environmental conditions could also be injurious to a cyclist's health [8]. Thus, this research exploited the Taiwan govermental open data platform [31] to provide environmental information to the cyclists via the mobile app.

B. System Development-Relevant Technologies

1) The Internet of things: Over the past decade, the term "Internet of things" has become one of the popular keywords among other emerging computer science technologies – such as big data analysis, cloud and edge computing, and deep learning – and these technologies usually work together to bring people new applications and services, especially when taking advantage of the valuable "big sensory data" collected by the Internet of things [37], [38]. The Internet of things extends from the buzzword "Internet" in a sense that it is an augmentation of the Internet – while the Internet connects people behind computers and smartphones, the Internet of things connects "things" which are upgraded to have computing and storage powers, sensing capabilities, and the ability to connect to the Internet. In the context of this research, the "thing" is the wearable device – the terminal – and the smartphone of the proposed system. The wearable device consists primarily of a microcontroller development board – the center of computing and storage powers, a BLE module – to connect to the user's smartphone, and pertinent physiological information sensors – to capture necessary sensory data the system requires. Besides, specific electrical circuits must be designed to appropriately connect these components, and the firmware must be programmed and written into the development board to make the wearable device operable. The wearable device does not connect to the Internet directly; rather, it first connects to the user's smartphone via BLE, and then indirectly connects to the Internet through the user's smartphone via mobile communications. This is how the proposed system can collect the physiological sensory data at the back-end server, exploiting the user's smartphone as a bridge (note that this is not implemented in the current proposed system and will be a future issue beyond the scope of this paper). Moreover, the smartphone itself provides the system with necessary data as well, such as the three-axis accelerometer data, the user's location information, and the false alarm feedback reported by the user.

2) Deep Learning: Over the course of the evolution of artificial intelligence (AI), researchers and scientists have realized that the best way to imitate a human brain is simply to make artificial intelligence operate more like a human brain – in a sense, to achieve the goal by biomimicry. The technology of deep learning [32] – being one form of machine learning – might be a result of serendipity along the line with such philosophy. A deep learning model is trained using a deep neural network, where the term "deep" means that more than one hidden layer is introduced in the neural network. Note that the construction of the neural networks itself is an imitation of the operation of a human brain, and that is where the term "neural network" came from. It was difficult to develop in the realm of machine learning in the past due to the lack of enormous computing and storage powers, but the constraint has been relieved over the past decade with the advancement of computer hardware. Another difficulty to appropriately train a machine learning model in the past was the lack of big data. However, this situation has also changed thanks to the current highly connected world, especially when the IoT technology came on the scene, which brings us tremendous sensory data from various sources. This research took advantage of a five-hidden-layer deep neural network to train the proposed bicycle crash model based on the collected data gathered by a data collection method this research designed and conducted – including two data collection mobile apps and two types of data collection experiments. A simple situation is to be detected during cycling – A crash occurs or not? – which means that the trained bicycle crash model is to deal with the problem of *classification*. With *supervised learning*, the proposed model learned how to assess a crash event with a certain accuracy (refer to Section V-C2), where *accuracy* is defined as (TP+TN) / (TP + TN + FP + FN) with TP, TN, FP, and FN represent true positive (the model predicts that a crash has occurred and the prediction is correct), true negative (predicting no crash and it is correct), false positive (predicting a crash but it is wrong), and false negative (predicting no crash but it is wrong) respectively in terms of confusion matrix [39].

III. SYSTEM OVERVIEW AND DESIGN

The proposed services system consists of three main parts: *wearable devices*, a *service mobile app*, and *back-end services*. The system architecture is illustrated in Fig. 1. First, the *wearable device* consists mainly of a microcontroller board, physiological sensors, and a BLE module. It detects the cyclist's physiological information, such as heart rate, oxygen saturation, and body temperature, and then transmit the information to the cyclist's smartphone via BLE connection. Second, the cyclist's smartphone must be installed with the *service mobile app* this research developed. With the aid of the built-in three-axis accelerometer, GPS module, BLE module, and mobile communications module of the smartphone, the mobile app offers the following three categories of services: (1) co-working with the back end as *the client side of the cycling team services*, as in Fig. 2, which include (a) cycling team creation, (b) off-the-team warning, (c) pre-crash warning, and (d) crash detection/reporting; (2) the *physiological status services*, which are responsible for accepting and displaying the physiological information as well as alerting the cyclist when his/her physiological status is going to be unsuitable for exercising; (3) the *environmental information service*, which informs the user of the current $PM_{2.5}$, air quality index (AQI), and ultraviolet (UV) index with the assistance of the Taiwan govermental open data platform [31]. Thirdly, the *back-end services* operate as the server side of the cycling team services and cover all the back-end support for the whole services, such as logging the crash locations (if any) of the cyclists (without recording any personal information) and collecting

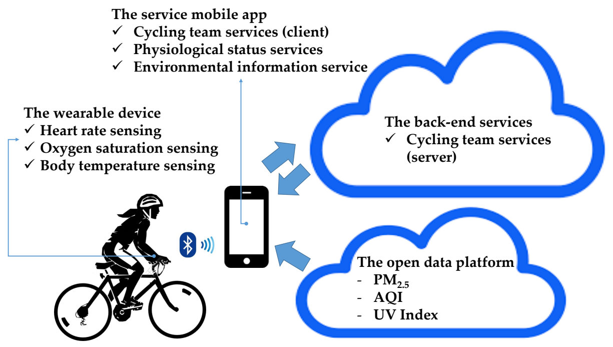

The service mobile app
✓ Cycling team services (client)
✓ Physiological status services
✓ Environmental information service
The wearable device
✓ Heart rate sensing
✓ Oxygen saturation sensing
✓ Body temperature sensing
The back-end services
✓ Cycling team services (server)
The open data platform
- $PM_{2.5}$
- AQI
- UV Index

Fig. 1. The system architecture.

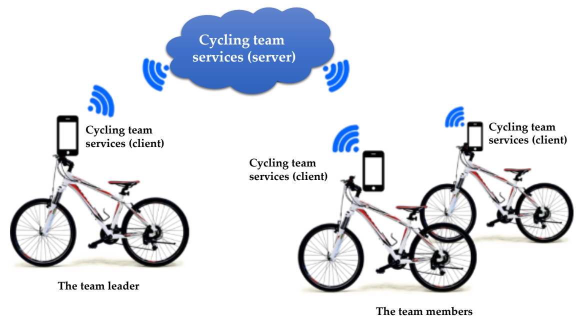

Cycling team services (server)
Cycling team services (client)
Cycling team services (client)
Cycling team services (client)
The team leader
The team members

Fig. 2. The illustration of the cycling team services.

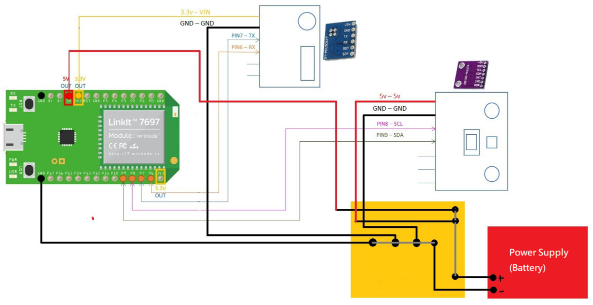


Fig. 3. The circuit diagram of the wearable device.

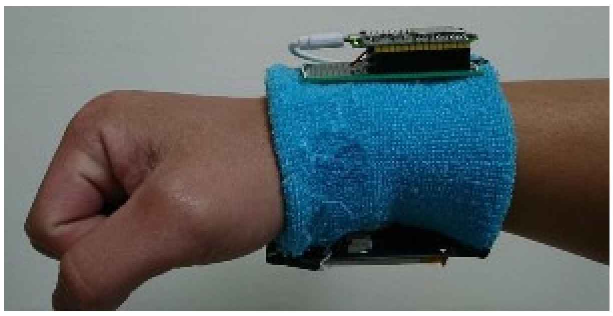


Fig. 4. The prototype of the wearable device.

the feedback for false crash alarms from the cyclists. Note that the Taiwan govermental open data platform [31] is a platform other than the back-end services this research developed.

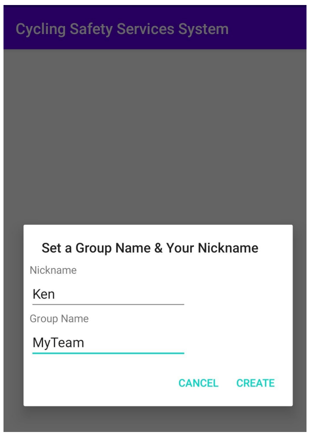
To implement the crash detection/reporting procedure in the mobile app, this research first built a deep learning bicycle crash model in advance for the procedure to assess whether a crash occurs. After the proposed system is commercialized and distributed to the consumers in the future, by continually collecting the false alarm feedback from cyclists, the crash model can be re-trained to lower its false alarm ratio, where the false alarm ratio is defined as the number of false alarms divided by the total number of alarms (including both false and correct alarms) – which can also be viewed as (1−precision) or [1−TP/(TP+FP)] in terms of confusion matrix [39], where TP stands for True Positive (the model perceives that a crash has occurred and the perception is correct) and FP indicates False Positive (the model perceives that a crash has occurred but the perception is wrong). It is also believed that the future collection and announcement of anonymous crash locations will be helpful to the cycling community because those locations could also be threatening to many other cyclists, where "anonymous" means that no personal information will be recorded together with those crash locations.

IV. System Implementation

This section elaborates on how this research implemented each main part of the proposed system, respectively.

A. The Wearable Device

This research used the LinkIt 7697 as the microcontroller development board for prototype development, where the term "prototype" is used from the perspective of the authors' cooperating IoT application vendor. This board contains a highly integrated system on a chip (SoC) – MediaTek MT7697 – designed for IoT application development, with Wi-Fi and BLE communications modules built-in. For future commercialization, other smaller and less expensive SoCs can be considered as a replacement. As for the heart rate and oxygen saturation sensors, this research chose the MAX30100 sensor solution and connected it to the LinkIt 7697 via an Inter-Integrated Circuit (I^2C) bus. Regarding the body temperature sensor, this research exploited the MLX90615 Infrared Thermometer and connected it directly to the serial port of the LinkIt 7697. The primary goal of the Arduino code written in the LinkIt 7697 is supposed to read the data detected by the sensors and to transmit the data via the mount-in BLE module of the LinkIt 7697. It is required to define the universal unique identifiers (UUIDs) for BLE services and characteristics on the board for BLE operation, in addition to adding the characteristics into related services respectively. Then the services have to be added into the Arduino library BLEPeripheral. After configuration, the physiological information can be placed into pertinent characteristics and sent to the smartphone via notifyAll( ) method. The circuit diagram is illustrated in Fig. 3 and the prototype of the wearable device is shown in Fig. 4.

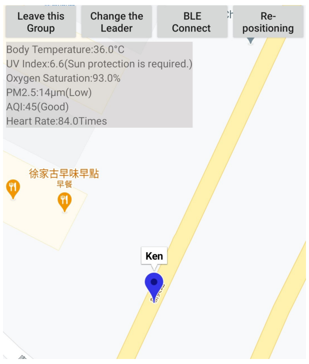

Cycling Safety Services System

Set a Group Name & Your Nickname

Nickname: Ken

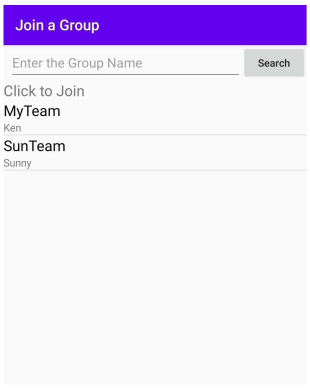
Group Name: MyTeam

Fig. 5. The mobile app function – cycling team creation: The team leader's nickname and group name configuration (cropped UI).



Body Temperature:36.0°C
UV Index:6.6(Sun protection is required.)
Oxygen Saturation:93.0%
PM2.5:14μm(Low)
AQI:45(Good)
Heart Rate:84.0Times

Fig. 6. The mobile app function – cycling team creation: The team leader (Ken)'s UI (cropped).


Join a Group

Click to Join

MyTeam
Ken

SunTeam
Sunny

Fig. 7. The mobile app function – cycling team creation: The (cropped) UI for a team member to select a group.

B. The Service Mobile App

This subsection describes how this research implemented the various functionalities of the service mobile app in respective subsubsections. It is noteworthy that the cyclists *do not* need to register to the proposed system. Therefore, *no* personal registration information will be stored in the proposed system, but only a temporary self-chosen nickname will be associated with each cycling team member. Also, to serve groups of cycling teams, the location information of each team member of a group will be tracked and broadcast to their own group, but the position tracking data will *not* be kept in the database, either.

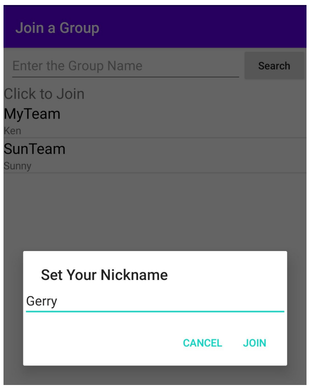
1) Cycling team services (Client): Cycling team creation: One can use the service mobile app to create a cycling team group (this person will be viewed as the team leader), and others who join the group afterwards will be regarded of as the team members. To create a cycling team group and take the responsibility of a team leader, the user needs to press "Create a Group" button and then configure his/her nickname (e.g., Ken) and the group name (e.g., MyTeam) as in Fig. 5 (the cropped user interface). Once the group is created, the app will turn to the leader's user interface (UI) as in Fig. 6 (the cropped UI), and the GPS positioning functionality will be activated. Note that the blue droplet on the map in the leader's UI expresses the leader's location. The leader can also switch the team leader role to another member in the group. The switch process is initiated by pressing "Change the Leader" button in the leader's UI.

After a group is created, each of the team members can press "Join a Group" button and then search and choose the group he/she wants to join as in Fig. 7 (the cropped UI). Then he/she can configure his/her nickname (e.g., Gerry) as in Fig. 8 (the cropped UI). Once this member joins a group, the app will turn to the team member's UI as in Fig. 9 (the cropped UI), and the GPS positioning functionality will be turned on. The green droplet (lighter in grayscale) illustrated on the map in the team member's UI points out this member's own location, whereas the leader's location is depicted as a blue droplet (darker in grayscale) on the map in the team member's UI.

2) Cycling Team Services (Client): Off-the-team warning: The mobile app uses the great-circle distance formula [40] to continually calculate the distance between each of the team members and the team leader (whose location is the benchmark), based on their longitudes and latitudes. If the distance is more than 25 meters, the mobile app will regard of this situation as a possibility of "off-the-team" and start to warn the off-the-team member (e.g., Jeff) as in Fig. 10 (the cropped UI) and all the other members in the team as in Fig.11 (the cropped UI). In the specific example shown in Fig. 10 and Fig. 11, Jeff has been too far away from the leader, and both Jeff and the team members got an off-the-team warning from the app.

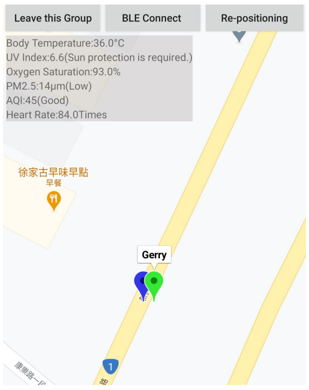
3) Cycling team services (Client): Pre-crash warning: This function exploits the built-in three-axis accelerometer of the user's smartphone to infer the angle of inclination of the bicycle's head or the handlebar, where the reference x-axis is the tangent line of the Earth's surface. Note that the user's smartphone should be fixed head up at the middle of the bicycle's head.

This research used a simple evaluation rule to judge the possibility of a crash (see Fig. 12): Define $angle_V$ as the angle between the vertical center line of the bicycle and the reference



Gerry

Fig. 8. The mobile app function – cycling team creation: The team member's nickname configuration (cropped UI).

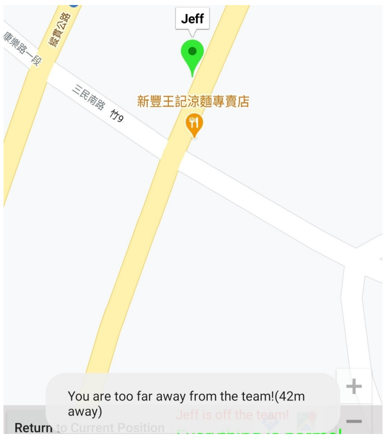


Body Temperature:36.0°C
UV Index:6.6(Sun protection is required.)
Oxygen Saturation:93.0%
PM2.5:14μm(Low)
AQI:45(Good)
Heart Rate:84.0Times

Fig. 9. The mobile app function – cycling team creation: The team member (Gerry)'s UI (cropped).

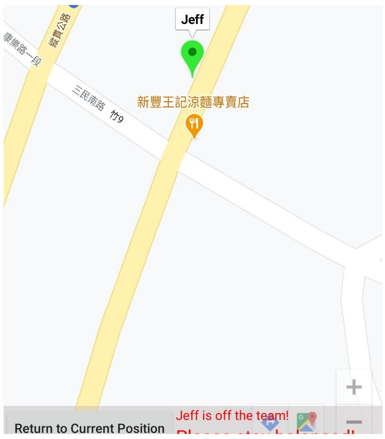
y-axis, which is the vertical line to the Earth's surface tangent. Whenever $angle_V$ is more than 45°, a crash could occur due to an imbalance. Note that this $angle_V$ is exactly equal to the angle, $angle_H$, between the handlebar and the reference x-axis. Then, whenever the $angle_H$ is growing closer to 45°, say, 30° as the threshold setting, the mobile app must alert the cyclist. With 30° as the threshold, the absolute threshold value of the smartphone's x-axis acceleration can thus be computed as "9.8 m/s^2 (the acceleration of gravity) times cos 60°," which is approximately 5 m/s^2. Therefore, any time the absolute value of the measured smartphone's x-axis acceleration is greater than 5 m/s^2, the mobile app perceives that the $angle_H$ is larger than the threshold 30° and will immediately issue the pre-crash warning to the cyclist, as in Fig. 13 (the cropped UI).

4) Cycling team services (Client): Crash detection/reporting: The mobile app can continually evaluate whether a crash occurs according to a pre-established bicycle crash model (refer to Section V) with the assistance of the mount-in three-axis accelerometer of the user's smartphone. Once the mobile app considers that a crash event takes place, the services system will automatically broadcast this event to all the other members of the team, prompting them with an SOS text message – including the crashed member's



You are too far away from the team!(42m away)

Fig. 10. The mobile app function – off-the-team warning: The off-the-team member (Jeff)'s UI (cropped) with "You are too far away from the team! (42 m away)" indicated at the bottom.



Jeff is off the team!

Fig. 11. The mobile app function – off-the-team warning: other members' UI (cropped) with "Jeff is off the team!" indicated at the bottom.

nickname (e.g., Michael) – as well as a marked red droplet on the map to pinpoint the crashed member's location, as in Fig. 14 (the cropped UI). As for the crashed member, his/her app will pop up a dialog as in Fig. 15 (the cropped UI). If there is actually no crash occurring this time, the user can click "MISJUDGMENT" to stop announcing the SOS message and report this false alarm to the back end. After the proposed system is commercialized and distributed to the consumers in the future, the crash model can be re-trained based on all the false alarm reports gathered in the database. Otherwise, any person who comes to rescue the crashed member can click "YES" to stop the SOS announcement, or "NO" to switch off the dialog but let the SOS announcement keep running.

5) Physiological Status Services: This bunch of services displays the user's physiological information, including his/her body temperature, oxygen saturation, and heart rate, which are received from the user's wearable device via the internal BLE module of the user's smartphone. If any of the data is considered to be abnormal (refer to Section II), this bunch

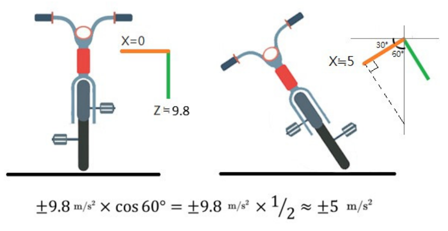


Fig. 12. An illustration of the evaluation rule to judge the possibility of a crash.

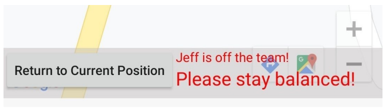
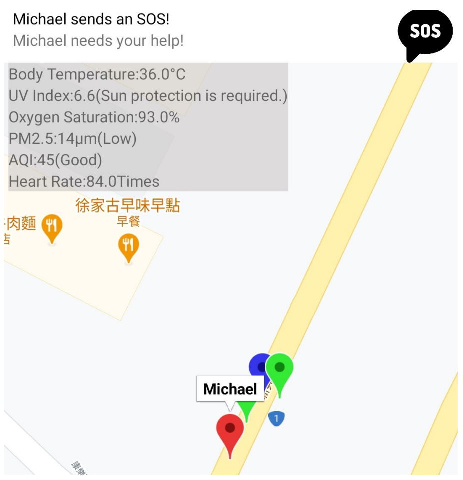


Fig. 13. The mobile app function – pre-crash warning (cropped UI): with "Please stay balanced!" indicated at the bottom.


Michael sends an SOS!
Michael needs your help!

Body Temperature:36.0°C
UV Index:6.6(Sun protection is required.)
Oxygen Saturation:93.0%
PM2.5:14μm(Low)
AQI:45(Good)
Heart Rate:84.0Times

Fig. 14. The mobile app function – crash detection/reporting: other members' UI (cropped) with "Michael sends an SOS! Michael needs your help!" indicated at the top.

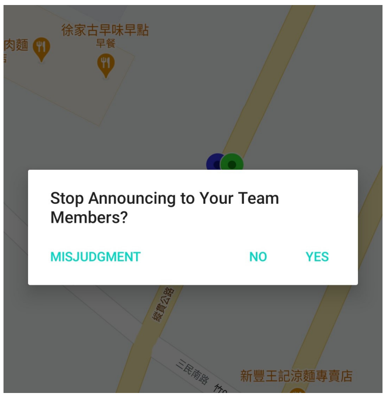

Stop Announcing to Your Team Members?

Fig. 15. The mobile app function – crash detection/reporting: The crashed member's UI (cropped).

of services will immediately alert the user. The physiological information is exhibited as in Fig. 16 (the cropped UI).

6) Environmental Information Service: This service takes advantage of the Taiwan govermental open data platform [31] to display several environmental indexes such as UV Index, $PM_{2.5}$, and AQI for the user. The mobile app communicates with the platform via HttpURLConnection. The indexes are demonstrated as in Fig. 16 (the cropped UI), too.

C. The Back-End Services

The back-end services cooperate with the service mobile app to provide the following functions: (1) Cycling team creation, (2) member location information collection and broadcasting, (3) member off-the-team broadcasting, (4) member crash event broadcasting, (5) crash location recording, and (6) false crash alarm report collection. As in Fig. 17, the prototype was implemented on a Linux-based NGINX web server with MariaDB as the database and PHP to access MariaDB, and the service mobile app interacted with the back-end services via HTTP, where the requests and responses were formatted in JSON data format.

V. The Bicycle Crash Model

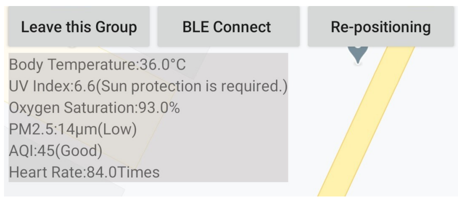
A. Introduction and Data Collection

Before the service mobile app can detect a crash event automatically, a bicycle crash model must first be built and then a detection procedure based on this crash model must be implemented in the mobile app. To build up this crash model, this research first designed a *data collection method* and implemented two data collection mobile apps to conduct data collection experiments. Then this research exploited statistical analysis and deep learning approaches respectively to build the model using the collected data. Finally, this research compared the results of the two approaches and decided to use the deep learning crash model to implement in the service mobile app.

The goal of the data collection experiments was to collect the three-axis accelerometer data with relevant metadata corresponding to "bicycle crash events" and "sudden bicycle motions without crashing" respectively. First, the two data collection mobile apps were installed on two smartphones respectively, where one smartphone was mounted at the middle of the handlebar with its head up to be the *collector* as in Fig. 18, and another smartphone – as the one held by the person in Fig. 19 – was exploited by the research as the *controller* to remotely control the *collector* app. Note that the repetitive data collection experiments were not conducted by simulation but carried out by actual bicycle moving, where at least two experimenters participated in the experiments – one was responsible for making the bicycle move, producing bicycle crashes, or causing the bicycle to perform certain kinds of motions (experimenter *M*) whereas another was responsible for remotely observe, control the *collector* app, and record necessary metadata (experimenter *R*). Since this preliminary research focused primarily on "recreational, amateur road biking

TABLE I
DATA COLLECTION EXPERIMENT *Types* AND *Subtypes*.

Type	Description	Subtype	Description
1	Bicycle crash events (The bicycle crashes during each Type 1 experiment).	1-1	Left crash
		1-2	Right crash
2	Sudden bicycle motions without crashing (The bicycle does not crash but just performs one of the five sudden motions during each Type 2 experiment).	2-1	Speed-up
		2-2	Emergency brake
		2-3	Left turn
		2-4	Right turn
		2-5	U turn



Body Temperature:36.0°C
UV Index:6.6(Sun protection is required.)
Oxygen Saturation:93.0%
PM2.5:14μm(Low)
AQI:45(Good)
Heart Rate:84.0Times

Fig. 16. The mobile app function – the physiological status services and the environmental information service (cropped UI).

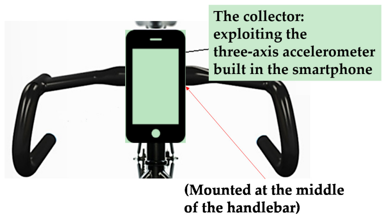


Fig. 18. The *collector* and the *controller*: The *collector* mounted at the middle of the handlebar, head up.

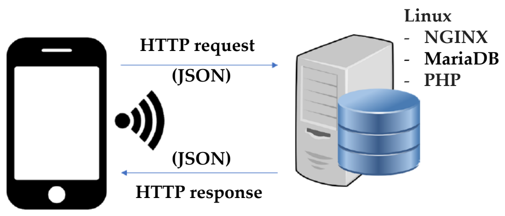


Fig. 17. The prototype of the back-end services.

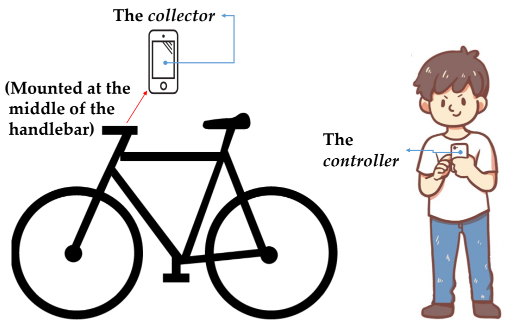


Fig. 19. The *collector* and the *controller*: The *collector* and the *controller* which is held by the person.

with a team" and "single-bicycle crashes" [16]–[18] only, not yet including racing, mountain biking, etc., the experiments were all carried out on flat, clement rooftops in the campus. Other crash scenarios – such as flip-over or collision with other road users – can be future research topics. Moreover, because this research did not have enough funds to create a fully safe environment for experimenter M to cause bicycle crashes by actually riding a bicycle during experiments without any possibility for experimenter M getting hurt, this research designed an alternative way for experimenter M to cause bicycle crashes which were only near-real crash experiments (refer to the following explanation). If in the future it is possible for the research to build a fully secure experimental environment, then real-crash experiments can be conducted using the same data collection mobile apps developed by this research and real-crash data can be collected to improve the bicycle crash model.

The main interfaces of the data collection mobile apps are shown in Figs. 20 and 21 (the cropped UIs). At first, experimenter R must press the "BLE Connect" button as in Fig. 20 at the *controller* side to connect the two smartphones via BLE. Then the *collector* app will automatically show the status page as in Fig. 21. From then on, this research conducted the data collection experiments – categorized in two types as in Table I – for multiple times.

Type 1 experiment – during which experimenter M must purposely produce a bicycle crash – was designed as Table II and is described in detail as follows, too.[1]

- Step 1. At the *controller* side, experimenter R must press "Start Collecting Data" button to notify the *collector* app to start gathering data when he sees that the bicycle begins to move. At the moment, the *collector* app starts collecting the three-axis accelerometer data with time and a status digit "0" every 100 ms. As for the status digit, this research defines the bicycle status in this phase as "Not Crashing (0)" for the bicycle is normally moving, so every record of the (x, y, z)-axes acceleration data collected before the next phase is marked with this status (i.e., 0).

[1]Note that to protect experimenter M's safety, bicycle crashes during Type 1 experiments are *not* produced by actually riding the bicycle but by an alternative way as described in Steps 1 and 2.

TABLE II
TYPE 1 EXPERIMENT DESIGN.

Step	Experimenter *M* / The bicycle	Experimenter *R* using the *controller* app	The *collector* mounted on the handlebar
1	Use a rope to drag the bicycle to move.	Press "Start Collecting Data" button when seeing that the bicycle begins to move.	Record the three-axis accelerometer data with time and a status digit "0" every 100 ms.
2	Let the bicycle naturally fall down to left or right due to any imbalance.	Press "Bicycle Starts to Crash" button when seeing that the bicycle begins to crash.	Continue to record the three-axis accelerometer data with time and a status digit "1" every 100 ms.
3	(The bicycle hits the ground.)	Press "Bicycle Completes Crashing" button when seeing that the bicycle hits the ground.	Continue to record the three-axis accelerometer data with time and a status digit "0" every 100 ms.
4	–	(1). Press "Stop Collecting Data" button. (2). Manually record Type and Subtype – whether the bicycle crashes to left or right – for *this round of experiment*.	Stop collecting data.

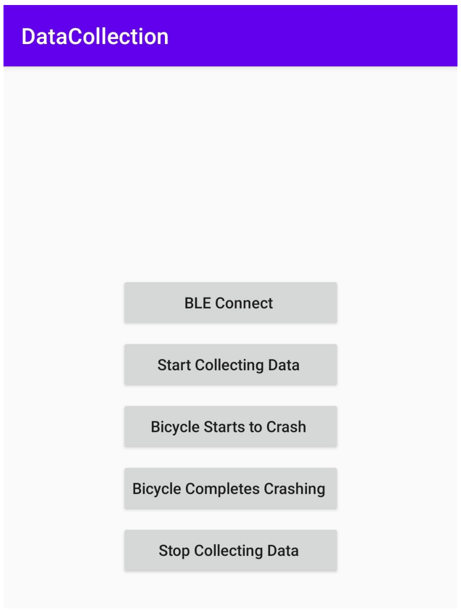


Fig. 20. The *data collection* mobile apps: The *controller* app UI (cropped).

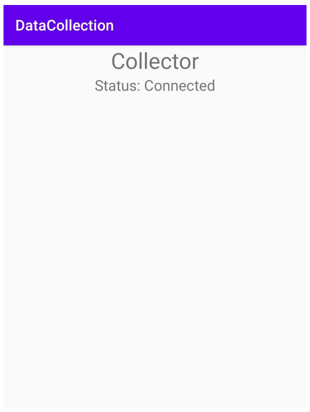


Fig. 21. The *data collection* mobile apps: The *collector* app UI (cropped).

- Step 2. When experimenter *R* sees that the bicycle begins to crash, he must immediately press "Bicycle Starts to Crash" button at the *controller* side to inform the *colletor* app to mark the subsequent collected data with status "1" because this research defines the bicycle status in this phase as "Crashing (1)" for the bicycle starts the crashing motion.
- Step 3. When experimenter *R* sees that the bicycle hits the ground, he must press "Bicycle Completes Crashing" button at the *controller* side to instruct the *collector* app to mark the following recorded data with status "0" because the bicycle status is also defined as "Not Crashing (0)" when the crash is over.
- Step 4. Finally, experimenter *R* must press "Stop Collecting Data" button and now a discrete-time Type 1 data series is obtained. Note that experimenter *R* must also manually record Type and Subtype (see Table I) – whether the bicycle crashed to left (subtype 1-1: Left crash) or to right (subtype 1-2: Right crash) – by *visual observation* for this round of Type 1 experiment.

Note that Type 1 experiment is meant to be conducted multiple times. After conducting Type 1 experiments over and over again, many sets of Type 1 discrete-time data series were gained for crash analysis afterwards. The related metadata included the timestamp and the bicycle status which were recorded along with the (x, y, z)-axes acceleration data. That is, each set of Type 1 data series contained a series of "(x, y, z)-axes accelerations, timestamp, and bicycle status" data, as illustrated in Fig. 22.

Now Type 2 experiment is explained – during which experimenter *M* must make the bicycle move in one of the five sudden ways without inducing crashes – as designed in Table III and the details are described as follows as well. Note that for the sake of convenience, when conducting Type 2 experiments, though the bicycle was not to crash but only to move in a certain sudden way (see Table I), this research used the same *controller* app UI to notify the *collector* app to mark data with status "1" from the beginning to the end of the sudden motion. Therefore, please do not be confused by the wording of the buttons pressed in Steps 2 and 3 of Type 2 experiment.

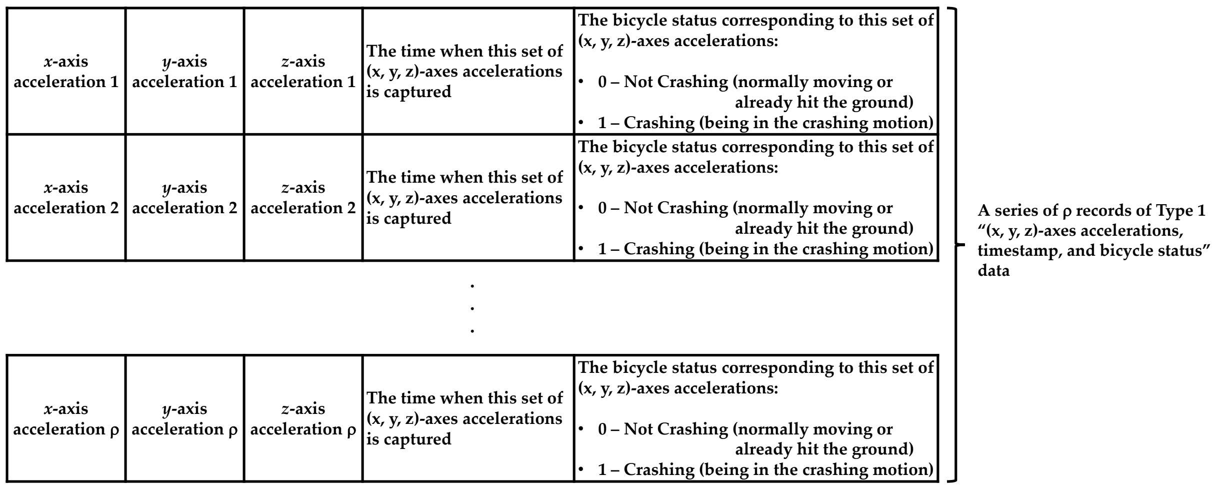

x-axis acceleration 1
y-axis acceleration 1
z-axis acceleration 1
The time when this set of (x, y, z)-axes accelerations is captured
The bicycle status corresponding to this set of (x, y, z)-axes accelerations:
• 0 – Not Crashing (normally moving or already hit the ground)
• 1 – Crashing (being in the crashing motion)
x-axis acceleration 2
y-axis acceleration 2
z-axis acceleration 2
x-axis acceleration ρ
y-axis acceleration ρ
z-axis acceleration ρ
A series of ρ records of Type 1 "(x, y, z)-axes accelerations, timestamp, and bicycle status" data

Fig. 22. One set of Type 1 discrete-time data series – which contains a series of "(x, y, z)-axes accelerations, timestamp, and bicycle status" data – obtained from one round of Type 1 experiment conduction.

TABLE III
TYPE 2 EXPERIMENT DESIGN.

Step	Experimenter *M* / The bicycle	Experimenter *R* using the *controller* app	The *collector* mounted on the handlebar
1	Ride the bicycle.	Press "Start Collecting Data" button when seeing that the bicycle begins to move.	Record the three-axis accelerometer data with time and a status digit "0" every 100 ms.
2	Perform one of the five sudden motions.	Press "Bicycle Starts to Crash" button when seeing that the bicycle begins one of the five sudden motions.	Continue to record the three-axis accelerometer data with time and a status digit "1" every 100 ms.
3	(The bicycle completes the sudden motion.)	Press "Bicycle Completes Crashing" button when seeing that the bicycle completes the sudden motion.	Continue to record the three-axis accelerometer data with time and a status digit "0" every 100 ms.
4	Stop riding.	(1). Press "Stop Collecting Data" button. (2). Manually record Type and Subtype of the bicycle sudden motion for *this round of experiment.*	Stop collecting data.

- Step 1. This step is the same as Step 1 in Type 1 experiment. At the *controller* side, experimenter *R* must press "Start Collecting Data" button to notify the *collector* app to start capturing data when he sees that the bicycle begins to move. At the moment, the *collector* app starts storing the three-axis accelerometer data with time and a status digit every 100 ms. Because this research defines the bicycle status in this phase as "Not During Sudden Motion (0)," the status digit of every record of the collected data is set as "0."
- Step 2. When experimenter *R* sees that the bicycle starts to move in one of the five sudden ways defined in Table I, he must press "Bicycle Starts to Crash" button[2] at once at the *controller* side to instruct the *colletor* app to mark the successive collected data with status "1" for this research defines the bicycle status in this phase as "During Sudden Motion (1)."
- Step 3. When experimenter *R* sees that the bicycle finishes the sudden motion, he must press "Bicycle Completes Crashing" button at the *controller* side to inform the *collector* app to mark the succeeding collected data with status "0" for the bicycle status goes back to "Not During Sudden Motion (0)" again.
- Step 4. Finally, experimenter *R* must press "Stop Collecting Data" button and now a discrete-time data series is acquired. Note that experimenter *R* must also manually record Type and Subtype (see Table I) – which subtype the bicycle performed: Speed-up (subtype 2-1), emergency brake (subtype 2-2), left turn (subtype 2-3), right turn (subtype 2-4), or U turn (subtype 2-5) – by *visual observation* for this round of Type 2 experiment.

Note that Type 2 experiment is also supposed to be carried out multiple times. After carrying out Type 2 experiments again and again, multiple sets of Type 2 discrete-time data series were ready for analysis later. The related metadata also

[2]Note that for the sake of convenience, when conducting Type 2 experiments, though the bicycle was not to crash but only to move in a certain sudden way (see Table I), this research used the same *controller* app UI to notify the *collector* app to mark data with status "1" from the beginning to the end of the sudden motion. Therefore, please do not be confused by the wording of the buttons pressed in Steps 2 and 3 of Type 2 experiment.

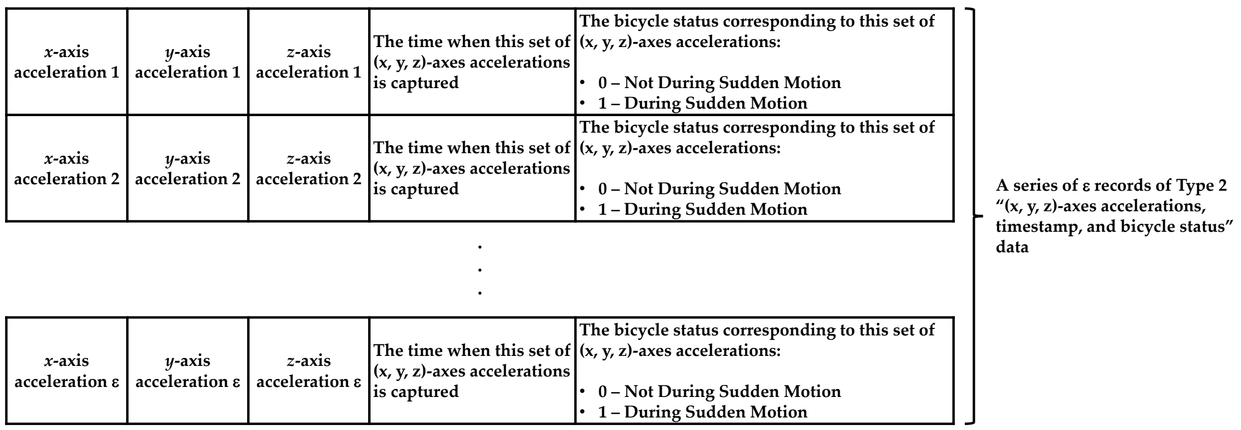

x-axis acceleration 1
y-axis acceleration 1
z-axis acceleration 1
The time when this set of (x, y, z)-axes accelerations is captured
The bicycle status corresponding to this set of (x, y, z)-axes accelerations:
• 0 – Not During Sudden Motion
• 1 – During Sudden Motion
x-axis acceleration 2
y-axis acceleration 2
z-axis acceleration 2
x-axis acceleration ε
y-axis acceleration ε
z-axis acceleration ε
A series of ε records of Type 2 "(x, y, z)-axes accelerations, timestamp, and bicycle status" data

Fig. 23. One set of Type 2 discrete-time data series – which contains a series of "(x, y, z)-axes accelerations, timestamp, and bicycle status" data – obtained from one round of Type 2 experiment conduction.

included the timestamp and the bicycle status recorded along with the (x, y, z)-axes acceleration data, which means that each set of Type 2 data series contained a series of "(x, y, z)-axes accelerations, timestamp, and bicycle status" data, too. Fig. 23 depicts one set of Type 2 data series acquired from one round of Type 2 experiment conduction.

The data analysis and model evaluation process was planned as in Fig. 24. After collecting all the experimental data series, this research preprocessed them by first calculating the magnitude of the net acceleration, $netAcc$, of every (x, y, z)-axes acceleration using (1):

$$netAcc = \sqrt{Acc_x^2 + Acc_y^2 + Acc_z^2}, \tag{1}$$

where Acc_x, Acc_y, and Acc_z represent the x-axis, y-axis, and z-axis accelerations respectively, and then selecting effective data for analysis. After data preprocessing, 133 valid and effective preprocessed data sequences, where *each* data sequence represented either "a crash occurred during bicycle movement (drawn from one set of Type 1 data series)" or "no crash occurred during bicycle movement (drawn from one set of Type 2 data series)" (refer to Section V-B for details). The 133 data sequences were divided into two parts: 92 data sequences were used as the training data, and 41 as the test data. As mentioned in the first paragraph of this subsection, this research analyzed the data in two ways: (1) Statistical analysis approach: This approach was exploited to find the threshold value right before a crash occurs. The implementation was done by Python programming and the data graphs were plotted using the matplotlib library. (2) Deep learning approach: This research fitted a bicycle crash model to the training data by Python programming with TensorFlow [41]. As a final step, this research used the test data to evaluate and compare the accuracies of the statistical analysis and deep learning crash models, where the *accuracy* is defined as (TP+TN) / (TP + TN + FP + FN) with TP, TN, FP, and FN represent true positive (the model perceives that a crash has occurred and the perception is correct), true negative (perceiving no crash and it is correct), false positive (perceiving a crash but it is wrong), and false negative (perceiving no crash but it is wrong) respectively in terms of confusion matrix [39]. The analysis and evaluation details are elaborated in the following subsections.

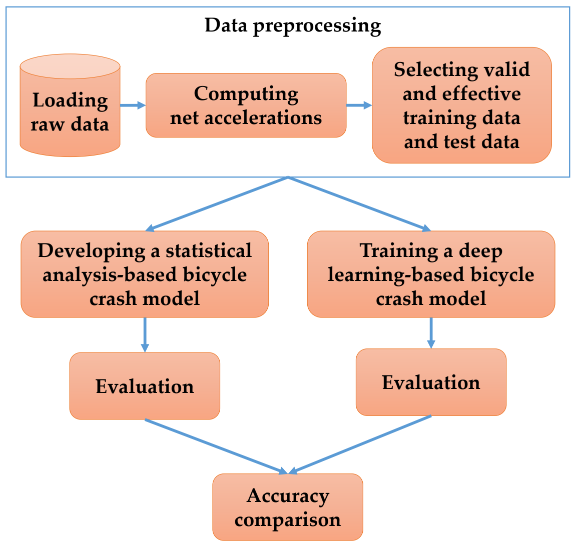
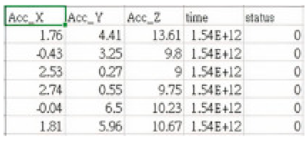

Data preprocessing
Loading raw data
Computing net accelerations
Selecting valid and effective training data and test data
Developing a statistical analysis-based bicycle crash model
Training a deep learning-based bicycle crash model
Evaluation
Evaluation
Accuracy comparison

Fig. 24. Data analysis and model evaluation process.

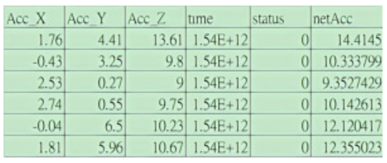
B. Data Preprocessing

As mentioned in Subsection V-A, the *collector* app recorded multiple sets of Type 1 and Type 2 discrete-time series of "(x, y, z)-axes accelerations, timestamp, and bicycle status" data. Each set of these experimental data series – as illustrated in Figs. 22 and 23 – was recorded as in Fig. 25 (a partial snapshot of the raw data stored in an Excel file), where the fields "Acc_X, "Acc_Y," and "Acc_Z" recorded the x-axis, y-axis, and z-axis accelerations respectively, and the fields "time" and "status" recorded the timestamp and the bicycle status.



Acc_X	Acc_Y	Acc_Z	time	status
1.76	4.41	13.61	1.54E+12	0
-0.43	3.25	9.8	1.54E+12	0
2.53	0.27	9	1.54E+12	0
2.74	0.55	9.75	1.54E+12	0
-0.04	6.5	10.23	1.54E+12	0
1.81	5.96	10.67	1.54E+12	0

Fig. 25. Part of the collected data (raw data) in one set of discrete-time data series.



Acc_X	Acc_Y	Acc_Z	time	status	netAcc
1.76	4.41	13.61	1.54E+12	0	14.4145
-0.43	3.25	9.8	1.54E+12	0	10.333799
2.53	0.27	9	1.54E+12	0	9.3527429
2.74	0.55	9.75	1.54E+12	0	10.142613
-0.04	6.5	10.23	1.54E+12	0	12.120417
1.81	5.96	10.67	1.54E+12	0	12.355023

Fig. 26. Part of the data (undergoing preprocessing) corresponding to Fig. 25, where the net accelerations (netAcc's) were already computed and appended.

Recall that the statuses 0 and 1 represent "Not Crashing" and "Crashing" respectively for Type 1 data, and "Not During Sudden Motion" and "During Sudden Motion" respectively for Type 2 data. The magnitude of the net acceleration, $netAcc$, was then computed by calculating the square root of (Acc_X^2 + Acc_Y^2 + Acc_Z^2) as equation (1) and appended into the table as in Fig. 26 (a partial snapshot of the data undergoing preprocessing).

Because the lengths of the discrete-time data series were *not* all the same, further effective data selection had to be done from these data series to make all the lengths equal so as to keep fairness and consistency. To do this, this research exploited data visualization to investigate these data series, for example, two of the data series plotted in Fig. 27. The data graphs were drawn according to data series like Fig. 26, where the data points with status "0" were marked in blue (darker in grayscale) whereas the data points with status "1" were marked in red (lighter in grayscale). The data visualization gave us the insight that *50 records of net accelerations within each data series were enough to evaluate a crash or a sudden motion*. Based on this insight, this research selected 133 valid data series out of all, where each of the 133 data series contained only 50 records of net accelerations along with the metadata (time and status). Note that this research purposely selected these 133 data series so that each one of the data series represents either "a crash occurred during bicycle movement" or "no crash occurred during bicycle movement." That is, the 133 data series are a mixture of Type 1 and Type 2 data series. The 50 records of net accelerations within each of the "a crash occurred during bicycle movement" data series were specifically extracted out of Type 1 data series *during a bicycle crash* according to the "status" field – data with status "1" indicate that they were collected exactly during a crash; those within "no crash occurred during bicycle movement" data series were specifically extracted from Type 2 data series *during a sudden motion* according to the "status" field, too – data with status "1" tell us that they were gathered precisely during a sudden motion. By doing so, it took less time to proceed with the succeeding data analysis as well.

Finally, this research (1) *copied* the 133 chosen data series, (2) *removed* the Acc_X, Acc_Y, Acc_Z, time, and status fields, (3) *merged* the 133 pruned sequences (each containing only 50 net accelerations) into one table, (4) *labeled* each sequence based on the experiment type of each sequence (see Table I) – recall that the *Type* was manually recorded by experimenter *R* at Step 4 in each round of experiment conductions (see Tables II and III) – label "1" indicates that the sequence was drawn from Type 1 data series and represents "a crash occurred during bicycle movement"; label "0" points out that the sequence originated from Type 2 data series and stands for "no crash occurred during bicycle movement." The result is partially exhibited in Fig. 28 (a partial snapshot of the table of the preprocessed data stored in an Excel file). To be specific, each row in the table in Fig. 28 represents one of the 133 sequences, so each row in the table contains 50 net accelerations and one label. Fig. 29 is a conceptualization of the preprocessed data. This table of 133 preprocessed data sequences was used in both the statistical analysis and deep learning approaches later.

C. Data Analysis

This research tried to gain more insights by the following exploratory data analysis and visualization. First, this research investigated the Type 1 graphs of the (x, y, z)-axes accelerations by time respectively, for instance, the three graphs corresponding to one of the raw data series (narrowed down to 50 records of data during a crash each) as in Fig. 30. Also, this research looked into the Type 1 graphs of the net accelerations by time, for example, the graphs shown in Fig. 31. Note again that the data with status "Crashing (1)" were all marked as red data points (lighter in gray scale), and those with status "Not Crashing (0)" were all marked as blue data points (darker in gray scale).

Moreover, this research studied the graphs corresponding to subtypes 1-1 (left crash) and 1-2 (right crash) respectively, for instance, the two graphs illustrated in Fig. 32. Furthermore, this research examined the graphs corresponding to subtypes 2-1 through 2-5, for instance, the five graphs shown in Fig. 33, where the blue (darker in grayscale) data points indicate that they were marked with status "Not During Sudden Motion (0)" and the red (lighter in grayscale) data points hint that they were marked with status "During Sudden Motion (1)."

Taking advantage of data visualization on Fig. 27 and Fig. 30 to Fig. 33 – specifically, noticing and comparing *the scales and the maximum numbers on the y-axis* of Type 1 graphs with those of Type 2 graphs, this research obtained the insight that Type 1 *variations* of accelerations were generally larger than Type 2 *variations* of accelerations. According to this knowledge, this research tried to use statistical analysis to find a certain threshold value right before a bicycle crash based on the different acceleration variations of Type 1 and Type 2 preprocessed data (see Section V-C1). Also, this research exploited the preprocessed data to train a bicycle crash model with a deep neural network classifier (see Section V-C2). The comparison of the results of these two approaches will be discussed in Section V-C3.

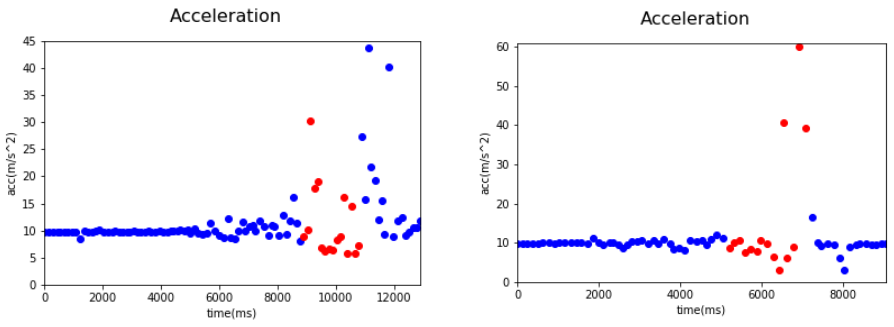
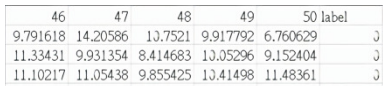


Fig. 27. Two Type 1 instances of the net accelerations by time.



46	47	48	49	50	label
9.791618	14.20586	10.7521	9.917792	6.760629	0
11.33431	9.931354	8.414683	10.05296	9.152404	0
11.10217	11.05438	9.855425	10.41498	11.48361	0

Fig. 28. Part of the *preprocessed data*.

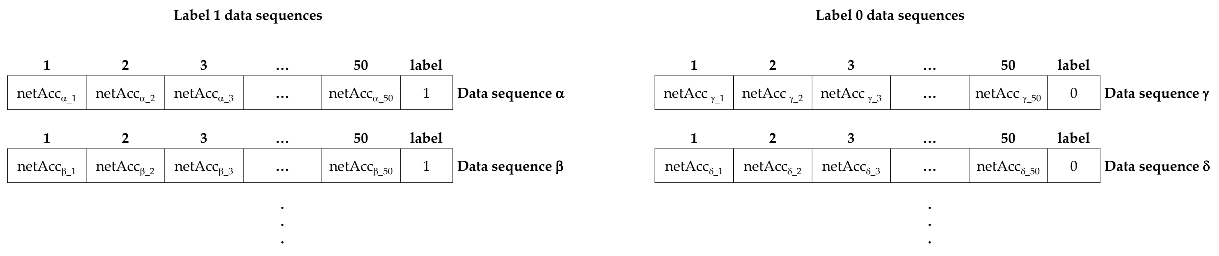
1) Statistical Analysis Approach: To verify the insight that statistical analysis could be used to find a certain bicycle crash threshold value by using the distinction between Type 1 and Type 2 acceleration variations, this research first separated the training data (92 data sequences out of the 133 preprocessed ones) into two groups according to the label field (a crash occurred during bicycle movement – “1” vs. no crash occurred during bicycle movement – “0”), and then calculated label 1 and label 0 average differences – μ_1 and μ_0 – between the maximum and minimum net accelerations respectively, as depicted in Fig. 34. This research observed that label 1 average difference – μ_1, was 48.65986 m/s^2, whereas label 0 average difference – μ_0, was 7.937981 m/s^2. The distinction between label 1 and label 0 average differences was quite obvious (just like the insight this research have gained about Type 1 and Type 2 net acceleration variations) and could be exploited to assess whether a bicycle crash occurs. Hence, this research further calculated label 1 and label 0 standard deviations (S.D.s) – σ_1 and σ_0 – of the differences between the maximum and minimum net accelerations respectively, as illustrated in Fig. 35. This research had the result that label 1 and label 0 standard deviations of the net acceleration differences – σ_1 and σ_0 – were 22.91112 m/s^2 and 5.581405 m/s^2 respectively. Therefore, it was concluded that (1) most of label 1 net acceleration differences were distributed within the range between $(\mu_1 \pm \sigma_1) = (48.65986 \pm 22.91112)$ m/s^2, and (2) most of label 0 net acceleration differences were distributed within the range between $(\mu_0 \pm \sigma_0) = (7.937981 \pm 5.581405)$ m/s^2. Based on this result, this research considered to use the middle value, 19.63406 m/s^2, between label 1 lower bound $(\mu_1 - \sigma_1) = (48.65986 - 22.91112)$ m/s^2 and label 0 upper bound $(\mu_0 + \sigma_0) = (7.937981 + 5.581405)$ m/s^2 to be the threshold value for evaluating a bicycle crash. To verify whether 19.63406 m/s^2 was a reasonable threshold value, this research tested different threshold values (7 m/s^2 to 49 m/s^2) on the training data to observe the accuracies as in Fig. 36, where the accuracy is defined as (TP+TN) / (TP + TN + FP + FN) in terms of confusion matrix [39].

From Fig. 36, this research recognized that the maximum accuracies fell at threshold values 18 m/s^2 and 19 m/s^2, which confirmed that 19.63406 m/s^2 could be a reasonable threshold value and this crash assessment approach could be feasible and effective. Consequently, this research used this threshold value (19.63406 m/s^2) to build a statistical analysis bicycle crash model, where the model is supposed to continuously calculate the difference between the maximum and minimum net accelerations within every 10 records of data consecutively (i.e., net acceleration data 1–10, then data 2–11, then data 3–12, etc.). If any difference is larger than the threshold value, the model will perceive that a crash occurs; otherwise, the model will identify that no crash occurs. *This research evaluated this model on the test data (41 data sequences out of the 133 preprocessed data sequences) and had an accuracy of 87.8048%.*

2) Deep learning approach: In addition to the statistical analysis crash model, this research also fitted a deep learning crash model to the training data using the deep neural network as in Fig. 37, where there were 50 input neurons, five hidden layers, and one output neuron. Within each hidden layer, there were 20, 50, 50, 20, and 10 neurons respectively. All the initial weights of the neural network were kept the same. The training process was conducted by DNNClassifier [42] in TensorFlow, where the iteration was set to 1500. Other pertinent hyper-parameters are listed as in Table IV. *This research evaluated the trained model on the test data and had an accuracy of 87.8049%.*

TABLE IV
THE CONFIGURATIONS OF OTHER HYPERPARAMETERS OF THE DEEP LEARNING MODEL.

Item	Configuration
Learning rate	0.1
Optimizer	Adagrad optimizer
Activation function	ReLU
Loss function	binary_crossentropy



Label 1 data sequences

1	2	3	...	50	label	
$netAcc_{\alpha\_1}$	$netAcc_{\alpha\_2}$	$netAcc_{\alpha\_3}$	...	$netAcc_{\alpha\_50}$	1	**Data sequence α**
$netAcc_{\beta\_1}$	$netAcc_{\beta\_2}$	$netAcc_{\beta\_3}$	...	$netAcc_{\beta\_50}$	1	**Data sequence β**
			. . .			

Label 0 data sequences

1	2	3	...	50	label	
$netAcc_{\gamma\_1}$	$netAcc_{\gamma\_2}$	$netAcc_{\gamma\_3}$	...	$netAcc_{\gamma\_50}$	0	**Data sequence γ**
$netAcc_{\delta\_1}$	$netAcc_{\delta\_2}$	$netAcc_{\delta\_3}$	...	$netAcc_{\delta\_50}$	0	**Data sequence δ**
			. . .			

Fig. 29. The conceptualization of the *preprocessed data*.

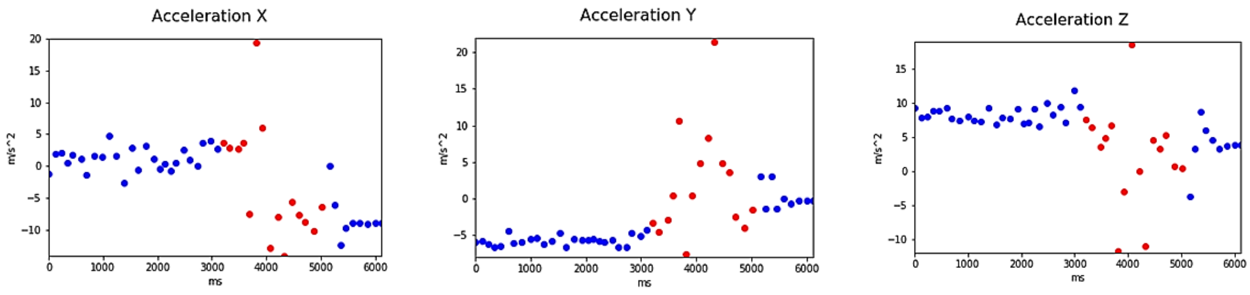


Fig. 30. A Type 1 instance of the (x, y, z)-axes accelerations by time respectively which have been narrowed down to 50 records of data during a crash each.

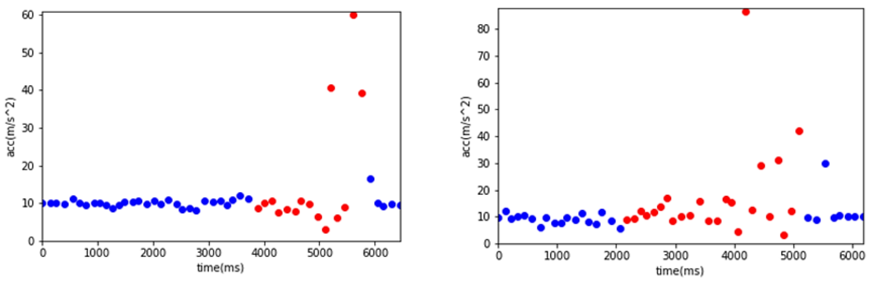


Fig. 31. Two Type 1 instances of the net accelerations by time which have been narrowed down to 50 records of data during a crash each.

3) Comparison between the statistical analysis and deep learning crash models: Compared the *accuracy* of the deep learning crash model – 87.8049% – with that of the statistical analysis crash model – 87.8048% – resulted from the test data, where the accuracy is defined as (TP+TN) / (TP + TN + FP + FN) in terms of confusion matrix [39], this research found that the difference was *marginal*. However, because the statistical analysis crash model using a threshold value is not as flexible as the deep learning approach whereas the deep learning model can keep improving by learning from more data and being fine-tuned on its hyperparameters and parameters after the proposed system is commercialized and distributed to the consumers in the future, it is believed that in the long run, the deep learning crash model could serve and perform better than the statistical analysis crash model. Consequently, this research decided to use the deep learning crash model in the service mobile app to detect a bicycle crash.

D. How the Deep Learning Crash Model Is Applied to Crash Detection

This research used the deep learning crash model to implement the crash detection/reporting procedure in the service mobile app. The crash detection/reporting procedure of the mobile app will continually collect the values of the three-axis accelerometer of the cyclist's smartphone and calculate the magnitude of the net acceleration. Every time the procedure obtains 50 records of the net acceleration (data 1–50, data 2–51, data 3–52, etc.), it will predict whether a crash occurs according to the deep learning crash model. Once a crash is regarded to occur by prediction, the service mobile app will send an urgent SOS message with the crashed member's location information to the back-end services, and the back-end services will broadcast an SOS message with this location information to the whole cycling team and pause crash assessment (the crash location will be recorded in the database as well). If this crash assessment is incorrect and the bicycle does

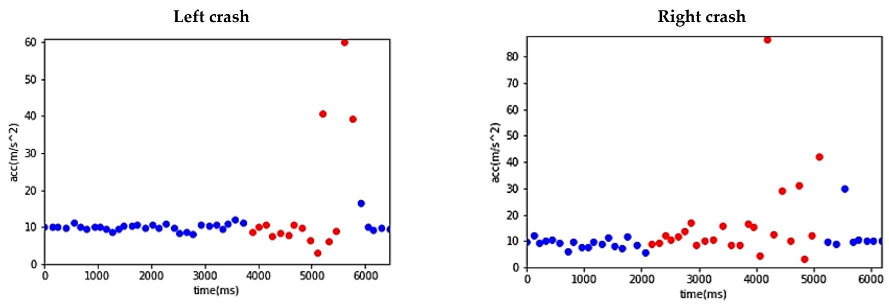


Fig. 32. A left crash and a right crash instances of the net accelerations by time which have been narrowed down to 50 records of data during a crash each.

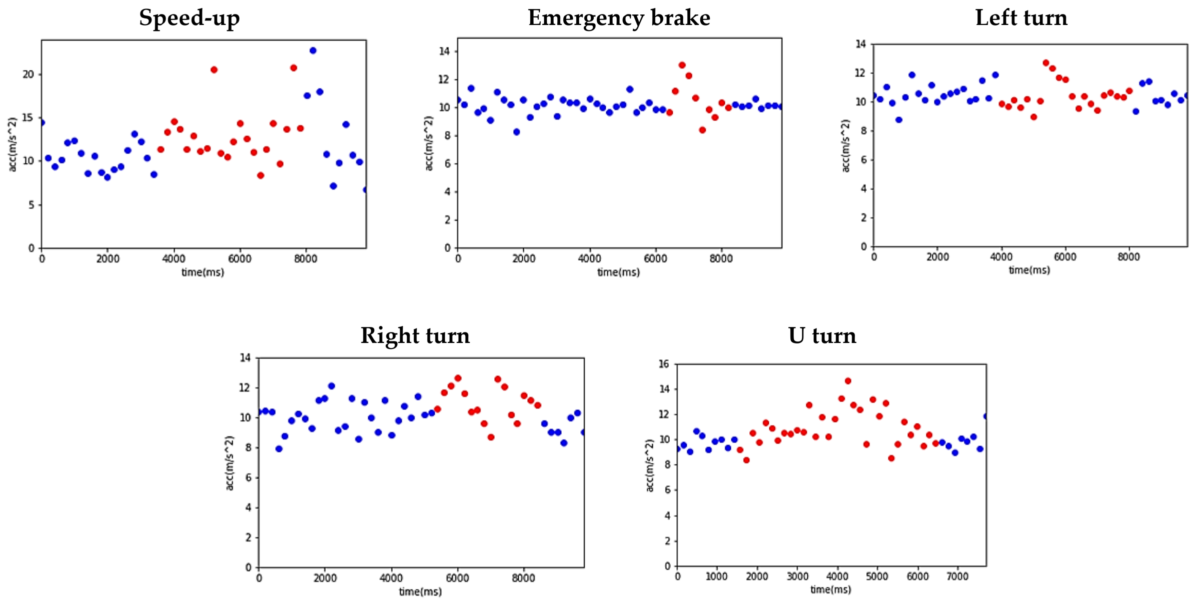


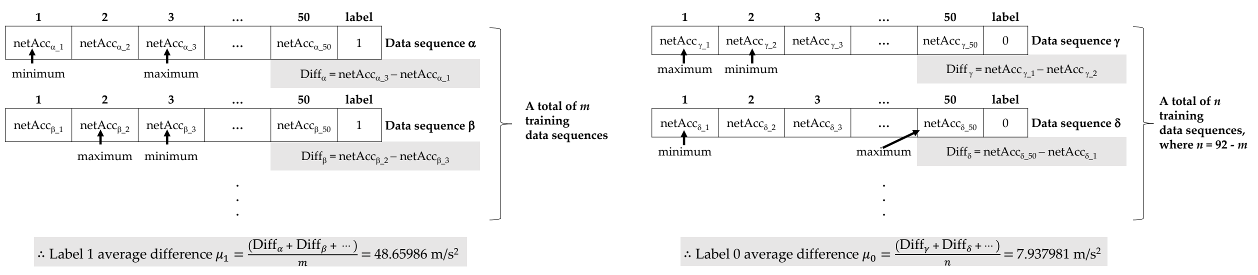
Fig. 33. Five Type 2 instances – each for a subtype defined in Table I – of the net accelerations by time which have been narrowed down to 50 records of data during a sudden motion each.

not crash actually, the cyclist can click "MISJUDGMENT" to report this false crash alarm to the back end and stop the SOS broadcast (see Fig. 15). After the proposed system is commercialized and distributed to the consumers in the future, the false alarm feedback (including the pertinent 50 records of data which caused this wrong prediction) from the cyclists can be exploited to re-train the deep learning crash model and keep lessening its false alarm ratio. Note that there may be intentional or unintentional false reports of "MISJUDGMENT"; therefore, if the accuracy of the re-fit model becomes worse, the re-fit model can just be abandoned and the previous version can continue to be in charge.

VI. Conclusion

Cycling has been popular for years around the world, especially in Taiwan in recent years, because it helps people improve their health condition. Even so, there are still certain potential dangers threating cyclists' safety. This research developed an IoT services system for cyclists (especially when they are cycling as a team) to resolve the threats – from terminal to back end – consisting of wearable devices, a service mobile app, and back-end services. With the aid of technologies such as deep learning model training and evaluation, IoT end device development (including circuit design to interconnect the BLE-embedded microcontroller board and the physiological information sensors as well as firmware programming), mobile app programming, RESTful API implementation, and open data utilization, this research developed the proposed *cycling safety services system* to provide the cycling team services, the physiological status services, and the environmental information service. The cycling team services – incorporating both the app side and the back end – primarily provide the functionalities of cycling team creation, off-the-team warning, pre-crash warning, and crash detection/reporting. Moreover, the back end is also to continually collect two sets of data: (1) The crash locations (if any) of the cyclists (not including any personal information) and (2) the feedback for false crash alarms reported by the cyclists after the system is actually distributed to the consumers.

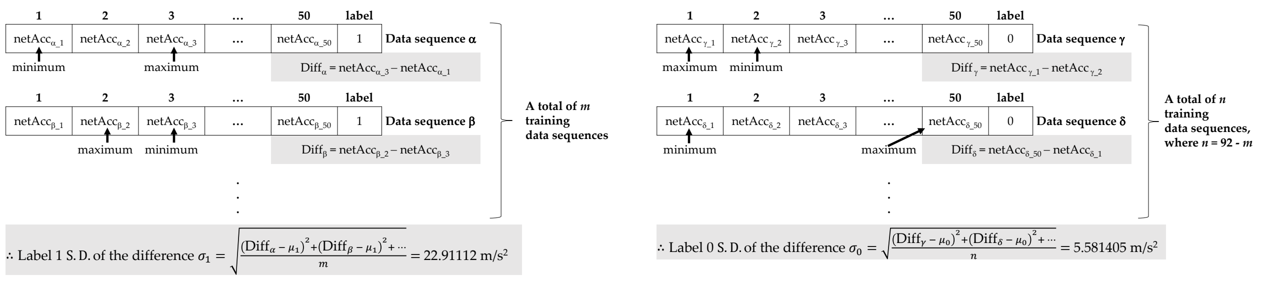
To enable the service mobile app to automatically detect


Data sequence α
Data sequence β
Data sequence γ
Data sequence δ
minimum
maximum
A total of m training data sequences
A total of n training data sequences, where n = 92 - m

$\therefore$ Label 1 average difference $\mu_1 = \frac{(\mathrm{Diff}_\alpha + \mathrm{Diff}_\beta + \cdots)}{m} = 48.65986$ m/s²

$\therefore$ Label 0 average difference $\mu_0 = \frac{(\mathrm{Diff}_\gamma + \mathrm{Diff}_\delta + \cdots)}{n} = 7.937981$ m/s²

Fig. 34. The label 1 and label 0 *average differences* between the maximum and minimum net accelerations respectively.


Data sequence α
Data sequence β
Data sequence γ
Data sequence δ
A total of m training data sequences
A total of n training data sequences, where n = 92 - m

$\therefore$ Label 1 S.D. of the difference $\sigma_1 = \sqrt{\frac{(\mathrm{Diff}_\alpha - \mu_1)^2 + (\mathrm{Diff}_\beta - \mu_1)^2 + \cdots}{m}} = 22.91112$ m/s²

$\therefore$ Label 0 S.D. of the difference $\sigma_0 = \sqrt{\frac{(\mathrm{Diff}_\gamma - \mu_0)^2 + (\mathrm{Diff}_\delta - \mu_0)^2 + \cdots}{n}} = 5.581405$ m/s²

Fig. 35. The label 1 and label 0 *S.D.s of the differences* between the maximum and minimum net accelerations respectively.

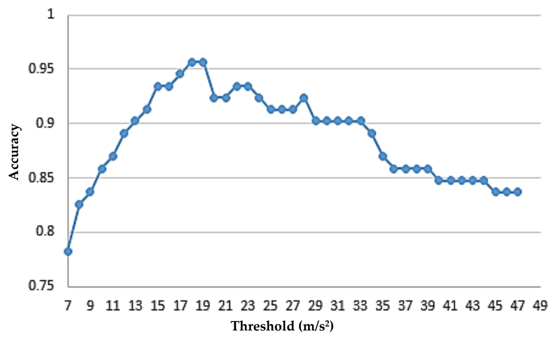


Fig. 36. The accuracies by different threshold values.

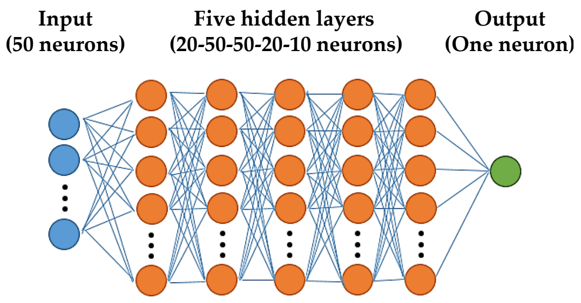


Fig. 37. The employed deep neural network architecture.

a bicycle crash, this research first conducted experiments to collect bicycle normal-motion/crash/sudden-motion data and then exploited these data to build two bicycle crash models in two approaches respectively: One was built based on statistical analysis and the other was fit using the deep learning approach. Curiously enough, the accuracies of the crash prediction based on these two models are almost the same: The statistical model achieves 87.8048% whereas the deep learning model reaches 87.8049%. Nonetheless, because the deep learning approach is supposed to be more flexible and can continue to improve by learning from more data and be fine-tuned on its hyperparameters and parameters, this research chose the deep learning model and implemented the crash detection/reporting procedure in the service mobile app based on the deep learning crash model.

Regarding the future works, first, the false alarm feedback gathered in the database after the system is distributed to the consumers can be utilized to re-train and enhance the crash model. Also, currently the re-training process is planned to be done manually and can be augmented later to be achieved automatically. In addition, the logged anonymous crash locations collected in the future can also be released to the cycling community to help raise an alert of risky regions. Furthermore, the future collection of the physiological information of the cyclists will be valuable for incubating new ideas and innovating new services, especially when the data amount grows big.[3] What is more, the prototype of the back-end services was implemented on a web server constructed by this research. To commercialize the proposed system in the future, the authors' cooperating IoT application vendor can

[3]Note that the personal data and privacy shall be seriously protected with every-aspect authorization, authentication, and accounting as well as data encryption and following national (and international, if distributed to the foreign markets) Computer-Processed Personal Data Protection Laws when being exploited by the authors' cooperating IoT application vendor after the proposed system is commercialized and distributed to the consumers. However, this topic is beyond the scope of this research, so it will not be discussed in this paper.

replace this web server with the commercial cloud solutions, such as Amazon AWS, Google Cloud Platform, Microsoft Azure, etc. To conclude, this research confirmed that the proposed system operates properly, and suggested that the proposed system, the deep learning bicycle crash model, the data collection method with its associated mobile apps, and the data collected during the experiments and in the future be beneficial to the cycling society as well as the researchers who conduct relevant research.

REFERENCES

[1] T. Götschi, J. Garrard, and B. Giles-Corti, "Cycling as a part of daily life: A review of health perspectives," *Trans. Rev.*, vol. 36, no. 1, pp. 45–71, 2016.

[2] P. Oja *et al.*, "Health benefits of cycling: A systematic review," *Scandinavian J. Med. Sci. Sports*, vol. 21, no. 4, pp. 496–509, Aug. 2011.

[3] M. P. Hoevenaar-Blom, G. C. W. Wendel-Vos, A. M. W. Spijkerman, D. Kromhout, and W. M. M. Verschuren, "Cycling and sports, but not walking, are associated with 10-year cardiovascular disease incidence: The MORGEN Study," *Eur. J. Cardiovasc. Prev. Rehabil.*, vol. 18, no. 1, pp. 41–47, Feb. 2011.

[4] J. Chertoff, "What's my ideal running heart rate?" [Online]. Available: [Online]. Available: https://www.healthline.com/health/running-heart-rate, 2018.

[5] J. D. Adams *et al.*, "Dehydration impairs cycling performance, independently of thirst: A blinded study," *Med. Sci. Sports Exerc.*, vol. 50, no. 8, pp. 1697–1703, Aug. 2018.

[6] Wikipedia, "Oxygen saturation (medicine)." [Online]. Available: https://en.wikipedia.org/wiki/Oxygen_saturation_(medicine), 2021.

[7] M. P. Suàrez-Mier and B. Aguilera, "Causes of sudden death during sports activities in Spain," *Rev. Esp. Cardiol.*, vol. 55, no. 4, pp. 347–358, Apr. 2002.

[8] D. McCarthy, "This deadly pollution must stop! London cyclists rally for health and climate." [Online]. Available: https://theecologist.org/2016/apr/20/deadly-pollution-must-stop-london-cyclists-rally-health-and-climate, 2016.

[9] Statistics Canada, "Circumstances surrounding cycling fatalities in Canada, 2006 to 2017." [Online]. Available: https://www150.statcan.gc.ca/n1/pub/82-625-x/2019001/article/00009-eng.htm, 2019.

[10] Ministry of the Solicitor General, Ontario, "Cycling death review." [Online]. Available: http://www.mcscs.jus.gov.on.ca/english/DeathInvestigations/office_coroner/PublicationsandReports/CyclingDeathReview/DI_Cycling_Death_Review.html, 2012.

[11] P. A. Cripton *et al.*, "Severity of urban cycling injuries and the relationship with personal, trip, route and crash characteristics: Analyses using four severity metrics," *BMJ open*, vol. 5, no. 1, p. e006654, 2015.

[12] M. Ohlin, B. Alguràn, and A. Lie, "Analysis of bicycle crashes in Sweden involving injuries with high risk of health loss," *Traffic Inj. Prev.*, vol. 20, no. 6, pp. 613–618, 2019.

[13] R. Buehler and J. Pucher, "The growing gap in pedestrian and cyclist fatality rates between the United States and the United Kingdom, Germany, Denmark, and the Netherlands, 1990–2018," *Trans. Rev.*, vol. 41, no. 1, pp. 48–72, 2021.

[14] L. De Rome *et al.*, "Bicycle crashes in different riding environments in the Australian capital territory," *Traffic Inj. Prev.*, vol. 15, no. 1, pp. 81–88, 2014.

[15] R. Aldred and A. Goodman, "Predictors of the frequency and subjective experience of cycling near misses: Findings from the first two years of the UK Near Miss Project," *Accid. Anal. Prev.*, vol. 110, pp. 161–170, Jan. 2018.

[16] P. Schepers *et al.*, "An international review of the frequency of single-bicycle crashes (SBCs) and their relation to bicycle modal share," *Inj. Prev.*, vol. 21, no. e1, pp. e138–e143, 2015.

[17] R. Utriainen, "Characteristics of commuters' single-bicycle crashes in insurance data," *Safety*, vol. 6, no. 1, Feb. 2020.

[18] B. Beck *et al.*, "Crash characteristics of on-road single-bicycle crashes: An under-recognised problem," *Inj. Prev.*, vol. 25, no. 5, pp. 448–452, Oct. 2019.

[19] "Intelligent Transport System (ITS); Vulnerable Road Users (VRU) awareness; Part 2: Functional Architecture and Requirements definition; Release 2. ETSI TS 103 300-2 V2.1.1," tech. spec., ETSI, 2020.

[20] M. R. Ibrahim, J. Haworth, N. Christie, T. Cheng, and S. Hailes, "Cycling near misses: A review of the current methods, challenges and the potential of an AI-embedded system," *Trans. Rev.*, vol. 41, no. 3, pp. 304–328, 2021.

[21] M. R. Ibrahim, J. Haworth, N. Christie, and T. Cheng, "CyclingNet: Detecting cycling near misses from video streams in complex urban scenes with deep learning," *arXiv preprint arXiv:2102.00565*, 2021.

[22] F. Tabei, B. Askarian, and J. W. Chong, "Accident detection system for bicycle riders," *IEEE Sensors J.*, vol. 21, no. 2, pp. 878–885, Jan. 2021.

[23] I.-H. Peng, P.-C. Lee, C.-K. Tien, and J.-S. Tong, "A physiological information indication and bicycle crash detection/reporting system for cycling (translated from Chinese)," in *Proc. TANET*, Sep. 2019.

[24] S. Joshi, A. Sabet, and D. Simcik, "Making cycling safer with AWS DeepLens and Amazon SageMaker object detection." [Online]. Available: https://aws.amazon.com/tw/blogs/machine-learning/making-cycling-safer-with-aws-deeplens-and-amazon-sagemaker-object-detection/, 2020.

[25] B. Williams, *Bicycle crash detection: Using a voice-assistant for more accurate reporting*, Master's Degree, University of Oregon, USA, Jun. 2018.

[26] M. Zhu, L. Han, F. Liang, C. Xi, L. Wu, and Z. Zhang, "A novel vehicle open door safety system based on cyclist detection using fisheye camera and improved deep convolutional generative adversarial nets," in *Proc. IEEE IV*, Jun. 2019, pp. 2195–2201.

[27] M. Pourhomayoun, "Automatic traffic monitoring and management for pedestrian and cyclist safety using deep learning and artificial intelligence," *Mineta Trans. Inst. Pub.*, Sep. 2020.

[28] M. Garcia-Venegas, D. Mercado-Ravell, and C. A. Carballo-Monsivais, "On the safety of vulnerable road users by cyclist orientation detection using deep learning," *Machine Vision Applications*, vol. 32, no. 5, pp. 1–17, Aug. 2021.

[29] U. Alvi, M. A. K. Khattak, B. Shabir, A. W. Malik, and S. R. Muhammad, "A comprehensive study on IoT based accident detection systems for smart vehicles," *IEEE Access*, vol. 8, pp. 122480–122497, Jul. 2020.

[30] "Bluetooth Specification Version 4.2," tech. spec., Bluetooth SIG, Inc., 2014.

[31] National Development Council, Taiwan (R.O.C.), "DATA.GOV.TW." [Online]. Available: https://data.gov.tw/en.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, MIT Press, 2016.

[33] I. Baldwin and M. Dunn, "How to Train Using Heart Rate Zones." [Online]. Available: https://www.rei.com/learn/expert-advice/how-to-train-with-a-heart-rate-monitor.html.

[34] Maxim Integrated Products, Inc., "MAX30100 - Pulse Oximeter and Heart-Rate Sensor IC for Wearable Health." [Online]. Available: https://datasheets.maximintegrated.com/en/ds/MAX30100.pdf, 2014.

[35] B. M. Popkin, K. E. D'Anci, and I. H. Rosenberg, "Water, hydration, and health," *Nutr. Rev.*, vol. 68, no. 8, pp. 439–458, Aug. 2010.

[36] Melexis, "Download datasheet for MLX90615." [Online]. Available: https://www.melexis.com/en/documents/documentation/datasheets/datasheet-mlx90615, 2013.

[37] L. Atzori, A. Iera, and G. Morabito, "The Internet of things: A survey," *Comp. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.

[38] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Comm. Sur. Tut.*, vol. 20, no. 4, pp. 2923–2960, Fourthquarter 2018.

[39] S. Narkhede, "Understanding Confusion Matrix." [Online]. Available: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62, 2018.

[40] Wikipedia, "Great-circle distance." [Online]. Available: https://en.wikipedia.org/wiki/Great-circle_distance.

[41] TensorFlow, "An end-to-end open source machine learning platform." [Online]. Available: https://www.tensorflow.org.

[42] TensorFlow, "tf.estimator.DNNClassifier." [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/estimator/DNNClassifier, 2021.

I-Hsuan Peng received the BS degree in Electronic Engineering from Chung Yuan Christian University, Taiwan, in 2001 and the MS degree in Electrical Engineering and PhD degree in Communication Engineering from National Central University, Taiwan, in 2003 and 2008, respectively. Currently, she is an assistant professor of the Department of Multimedia and Game Development in Minghsin University of Science and Technology. Her research interests include wired and wireless networks, cloud computing, and the Internet of things (IoT).

Pei-Chun Lee received the BS, MS, and PhD degrees in Computer Science and Information Engineering from National Chiao Tung University, Taiwan, in 1998, 2000, and 2006, respectively. She joined the Faculty of the Department of Information Management at Minghsin University of Science and Technology, Taiwan, in 2006, and currently serves as an assistant professor. Her current research interests include the analysis and design of innovative and practical application service systems based on the Internet of things (IoT) technologies, mobile app, and cloud services, the Bluetooth mesh and LoRa/LoRaWAN networks, as well as positioning technologies and algorithms such as LoRa geolocation.

Chen-Kang Tien received his BS degree in Computer Science and Information Engineering from Minghsin University of Science and Technology, Taiwan, in 2020. He is now studying in the graduate school of Communication Engineering at National Central University, Taiwan. His current research interests include artificial intelligence (AI) and integrated system development incorporating the technologies of the Internet of things (IoT), mobile app, and web.

Jyun-Sen Tong received his BS degree in Computer Science and Information Engineering from Minghsin University of Science and Technology, Taiwan, in 2019. He is now studying in the graduate school of Information and Computer Engineering at Chung Yuan Christian University, Taiwan. His current research interests include integrated system development incorporating the technologies of the Internet of things (IoT), mobile app, and cloud services.