Toward a Smart System for Pets

by

Van-Tien HOANG

THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT OF A MASTER'S DEGREE
WITH THESIS IN INFORMATION TECHNOLOGY ENGINEERING
M.A.Sc.

MONTREAL, MARCH 5, 2021

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Lucas Hof, Thesis Supervisor
Department of Mechanical Engineering at École de technologie supérieure

Mr. Rolf Wuthrich, Thesis Co-supervisor
Department of Mechanical, Industrial and Aerospace Engineering at Concordia University

Mr. Jean-Pierre Kenné, President of the Board of Examiners
Department of Mechanical Engineering at École de technologie supérieure

Mr. Ilyass Tabiai, Member of the jury
Department of Mechanical Engineering at École de technologie supérieure

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON FEBRUARY 26, 2021

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

I would like to thank the following people, without whom I would not be able to complete this research and would not make it through this master's degree. My supervisors, Professor Lucas Hof (ÉTS) and Rolf Wuthrich (Concordia), whose insight and guidance helps me to frame the work from the beginning. Also, without their friendly, thoughtful, and continuous involvement, the research would not make it into current states.

The team at BeOneBreed, the research's industrial partner, whose willingness and energy to answer and discuss any questions as well as re-direct them to the correct parties for information, and without whom I would not be able to start the work.

Finally, I also would like to thank many other people, even many that I do not interact with directly but their invisible important roles are what keeps everything connected and work smoothly.

And biggest thanks to my family, their constant support through the years is the greatest source of energy for me in life. I own you everything I got.

Vers un écosystème intelligent pour les animaux de compagnie

Van-Tien HOANG

RÉSUMÉ

Lorsque les propriétaires d'animaux prennent soin de leurs animaux de compagnie, cela suscite le besoin d'améliorer le bien-être et le bonheur de leurs animaux. Cela nécessite la capacité de contrôler une gamme de jouets intelligents pour animaux de compagnie et de surveiller les activités des animaux. En faisant cela, le propriétaire a la possibilité de reconnaître un comportement inattendu et d'influencer positivement le comportement des animaux de compagnie. Ces systèmes de jouets intelligents sont également utiles à des fins de dressage d'animaux de compagnie, qui pourraient être étendus à des systèmes plus intelligents utilisés pour encourager un comportement spécifique des animaux de compagnie. Actuellement, il existe plusieurs produits à des fins de surveillance. Cependant, ils ne disposent pas de fonctionnalités d'intégration telles que les capacités à intégrer dans les systèmes de maison intelligente actuels et n'ont pas la possibilité d'ajouter de nouveaux appareils au produit. Cette étude vise à proposer des systèmes multi-usages capables de suivre et d'enregistrer les données des animaux de compagnie (ex. Mouvements, température corporelle) et qui peuvent être intégrés dans d'autres systèmes intelligents. La thèse a mis en place une vitrine pour un tel système qui comprend un collier apposé sur l'animal, interagissant avec une application mobile, un stockage de données dans le cloud et une fonction analytique. Les résultats obtenus de l'étude présentée montrent qu'il est possible de concevoir et de mettre en œuvre l'architecture du système ciblé. Il démontre également les capacités d'utiliser des algorithmes d'apprentissage automatique pour détecter les comportements d'animaux.

Mots-clés: Internet des objets (IoT), architecture logicielle, animaux de compagnie, systèmes intelligents, classification du comportement des animaux de compagnie, applications mobiles, cloud computing

Toward a Smart System for Pets

Van-Tien HOANG

ABSTRACT

When pet owners take care of their pets, it sparks the need to improve the well-being and happiness of their pets. This requires the ability to control an array of smart toys for pets and monitor the animals' activities. By doing that, the owner has the possibilities to recognize unexpected behavior and to positively influence the behavior of the pets. Such smart toys' systems are also useful for pet training purposes, which could be extended to more intelligent systems used to encourage specific behavior of pets. Currently, there are several existing products for monitoring purposes. However, they are lacking integrating features such as capabilities to be incorporated within current smart home systems and lacking the ability to add new devices to the product. This study aims to propose multi-use systems that can track and record data of pets (e.g. movements, body temperature) and which can be integrated into other smart systems. The thesis has implemented a demonstration system that includes a collar affixed to the pet, interacting with a mobile application, a cloud data storage and an analytic function. The achieved results of the presented study show that it is feasible to design and implement the targeted system's architecture. It also demonstrates the abilities of using machine learning algorithms to detect pet behaviors.

Keywords: Internet-of-Things (IoT), software architecture, pets, smart systems, pet behavior classification, mobile applications, cloud computing

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF ABBREVIATIONS

BLE	Bluetooth Low Energy
IoT	Internet of Things
CSV	Comma-Separated Values
RDMS	Relational Database Management System
KNN	K-Nearest Neighbors
SVM	Support vector machine
UI	User Interface

LIST OF SYMBOLS AND UNITS OF MEASUREMENTS

A	Ampere
Mbps	Megabit per second
GB	Gigabyte
H	Hertz
J	Joule
m	Meter
mAh	Milliamp hour
MB	Megabyte
V	Volt
W	Watt

INTRODUCTION

Contexts and questions

Human has started living with pets for thousands of years (Clutton-Brock, 1999). Recent studies even found that Asian dogs were domesticated as far as 12,500 years ago and European wolves were domesticated for at least 15000 years ago (Frantz et al., 2016). This can be confirmed by Fig 0.1[1] which is a painting from an Egyptian tomb around 4000 years ago which depicts a human and a dog[2] together. The cat is also considered as pet from ancient time. The first evidence of house cats is found in Egypt as early as 4000 years ago (Barnes et al., 2018).

That long history of the relationship between human and pets create strong bonds between them (Serpell & Barrett, 2016; Myrick, 2015). That could be one of the reasons why there are more than 400 million domestic dogs worldwide as of today (Harari, 2014). Particularly, in the United States of America, there are more than 180 million dogs and cats according to American Pet Products Association's 2017-2018 National Pet Owners Survey (Association, 2019). This results in a huge expenditure for pets. For example, more than 69 billion dollars is spent on food, medical and other care for pets (see more at (Institute, 2019)).

When the owners take care of their pets, it leads to the need to know about their well-being. Commonly, a pet medical check-up is a choice. However, there is an ongoing need that pet-owners want systems to monitor their pet activities and could be able to recognize their behavior patterns. Besides these systems could be able to detect abnormal activities to alert and support the owner to take corrective measures (NCBI, 2019; McNicholas et al., 2005).

To this end, this thesis aims to investigate answers to the following research questions:

(Q1): How can we design the architecture of such a smart toy system?

---

[1] © Copyright Linda Evans/Australian Center for Egyptology, Macquarie University, Sydney

[2] https://www.livescience.com/59022-egypt-tomb-with-leashed-mongoose-drawing.html

Figure 0.1 An ancient drawing on the wall of a tomb at Beni Hassan in Egypt indicates a hunter holding the leash of a dog (bottom)

(**Q2):** Is it possible to implement such smart toy system architecture?

(**Q3):** Is it possible to provide useful information based on historical data of pet behaviours?

Research objectives

The project aims to create a smart toy system for pets based on Internet-of-Things (IoT) devices. Such systems can include a vast range of device types from a feeder, trackers to interactive toys for animals. By analyzing data collected from the devices (e.g., sensors, smart toys), the system goals are to improve human-animal interaction by understanding their pets' behaviors via connected products. Also, the system could help pet owners to decide the best actions to take according to the pet behavior. In order to do that, a behavior recognition application is proposed

and applies on collected data to predict the animal activities. Moreover, the system also includes a mobile application to displays the predicted pet behaviors and control certain functions of the system. Specifically, the research aims to:

(**A1):** Design system architectures for a smart toy system for pets.

(**A2):** Develop a demonstration system based on one system architecture.

In order to reach the thesis objectives, we propose a smart toy system for pets which can include:

- A range of connected devices allowing a better understanding of the animal's behavior. These devices can assist humans to identify the pet habits which can be used for a medical or behavioral diagnosis. Thus, the system can detect and record the events and activities of the animal.
- A software system that controls these devices. This software system is used to communicate between devices and performs data analysis to provide insights from pet activities.

The presented study was conducted in collaboration with the industrial partner BeOneBreed. Beonebreed is a young and innovative company in Beloeil, QC, Canada, and active in the field of pet-related products. They aim at creating innovative products with high-quality for pets, focusing on modern design and technical innovation implementation. Their overall objective is to generate a community for passionate pet owners to support them taking good care of their pets. It is important to note that in this research we utilize a hardware prototype (i.e., collar) which is provided by our industrial partner. It is also assumed that the hardware changes to be implemented towards product commercialization are taken over by a third party partner of BeOneBreed.

Thesis outline

The thesis is organized in 6 parts. *Introduction* chapter gives a brief overview of the conducted research studies. It describes the contexts and the thesis objectives. It also gives an overview of the proposed smart toy system for pets. Chapter 1 discusses the literature reviews. Chapter 2 provides the background knowledge and related technologies which are involved in building the system. Chapter 3 investigates the research methodologies and proposes potential system architectures. Chapter 4 describes the steps to build a demonstrated smart toy system. The final chapter, *Conclusion and Recommendation* summarizes the results, limits and provides possible future enhancements for the research.

CHAPTER 1

LITERATURE REVIEW

1.1 Introduction

This chapter is going to discuss related works on smart toy systems for pets. It describes relevant smart toys existed in the markets or in research and discusses common problems in this kind of system (Section 1.2). The chapter then examines the pet behavior analysis using wearable devices and its current methods in Section 1.3. In the end, there is a chapter summary.

1.2 Smart toys for pets

1.2.1 Commercial products

There are several smart commercial products for pets. They can be grouped into three main categories based on purposes: food-related controller, vital information monitor and geographical trackers.

In the food-related controller devices, Mookkie (Mookkie, 2019) is a typical product. It is a smart bowl identifying each pet visually (“face ID“) and associate them with a specific food. As a result, the diet can be customized for each pet. The system comes with a dedicated smartphone application providing notifications and short videos to the owners. However, this smart bowl does not interact with other "Mookkie" bowls or with other connected devices in the house.

Sure Petcare’s Pet Feeder Connect (Connect, 2020) is another product that helps the owner to achieve optimal pet diets . This product uses a scale to create a specific portion of food for pets. It also monitors the pet’s food consumption behavior. This system identifies each pet using a wearable tag for each animal. The feeder connects to a ’hub’ via a low energy protocol. The hub then uses a WIFI connection to communicate with the mobile phone, which in turn connects to

the internet. This product is designed for a single purpose and is not intended for adding new devices to this system.

In addition to diet control, measuring vital health-related data for pets is introduced in multiple products. Petpace (Petpace, 2019) is a product to monitor the pet well-being. It consists of a collar monitoring the dog's vital data (e.g., heart rate, body temperature) and alerts the owner when abnormality happens. Its architecture consists of three layers. The first layer contains the smart sensors affixed to the dog's collar. This sensor sends the actual data to the middle layer which is a base station. The base station communicates with the sensors and combines the data to send them to the cloud layer using the standard Ethernet interface. The calculation and analysis happen in the cloud layer. The owner can access those data wirelessly using their mobile application to consult their dog's Health Index, Activity Index, and Burned Calories. This product provides certain examples that inspire other pet well-being analysis products. Also, this collar does not provide the ability to connect with other toys and devices.

Similar to Petpace, Fit bark (FitBark, 2019) is a dog activity and sleep monitor device. It uses Bluetooth Low Energy (BLE) to make the connection with a WiFi base station. This technology promises lower energy consumption during data transmission, extending the lifetime of the battery in the collar. Any Android or iOS operated smartphone can be used as a base station.

For geographical locating purposes, Scollar (Scollar, 2019), is a dog tracking solution that provides not only information about the dog's activities but also offers tracking functions via GPS (Global Positioning System). Its system contains a base station (e.g. a smartphone) including a WiFi connection. Data collection and transfer technology are the same as in the previous devices (e.g., using Bluetooth). The main difference is that Scollar is the only device that provides a GPS sensor and an online dog tracking service. The collar also contains a temperature sensor and an integrated battery, with an estimated lifetime of 45 up to 60 days.

Moreover, there are multiple products that only provides location tracking function and an accompanied mobile application such as Fi Smart Dog Collar[1], Whistle Go Explore [2] and Garmin TT 15 GPS Dog Tracker Collar[3]. Their use cases are similar to that of Scollar.

These products show the current trend of smart devices for pets. They are all useful and provide a dedicated function. However, they are operating in a closed system and do not provide a standard way to add more connected things.

1.2.2 Research studies

In addition to these commercialized products, there are multiple academic studies about wearable devices for animals. Most of these are to monitor animal's activities for tracking or medical reasons. Pet Buddy (Ahn et al., 2016) is a device for dogs that aims to monitor the physical behavior of pets. The device is based on an Arduino main board, including gyroscope sensors. The collected sensor data are later transferred via Bluetooth to another Arduino-based logic board, which itself is connected to a computer to analyze the canine behaviors. Similar to Pet Buddy, WagTag (Weiss et al., 2013; Ladha et al., 2013b; Kumpulainen et al., 2018) are collar-based solution to track dogs' activities which uses Bluetooth to transfer data to the computer.

Instead of monitoring basic physical activities, David Sec et. al (Sec et al., 2018b) propose a system to monitor the health condition of dogs to detect possible epileptic seizures. This wearable device has a noise sensor to identify breath frequency, and it includes a heartbeat sensor. It exchanges data via BLE 4.0 with a Raspberry running Windows IoT operating system. The collected data are sent to a processing server via FTP.

Besides domestic pets (e.g., dogs, cats), there are similar systems for monitoring bigger animals such as pigs or cows. (Haladjian, Ermis, Hodaie & Brügge, 2017) creates iPig – a system

[1] https://tryfi.com/

[2] https://www.whistle.com/products/whistle-go-explore-gps-pet-tracker-activity-monitor

[3] https://buy.garmin.com/en-US/US/p/160889

to monitor free-roaming pigs via GPS and cellular data. This system consists of many small devices which are attached to the pig's ear. Furthermore, (Haladjian, Haug, Nüske & Bruegge, 2018) proposes a system which consists of wearable devices affixed to one leg of the cow to measure its abnormal posture.

The similarities of the aforementioned works are the isolation of the wearable devices: they are not designed to be integrated with external services or operate in a network with other connected IoT devices.

One example which demonstrates the possible interactions between multiple devices is from Y. Guo et. al (Guo et al., 2006). They prototype a wireless sensor platform for livestock (i.e., cows) behavior tracking. The board has GPS receivers and temperature, magnetometer, and accelerometer sensors. The nearby animals of the herd form a wireless sensor network to relay the data to the central server. However, due to the integration of multiple sensors and functionalities, the device is too heavy for domestic pets. Also, domestic pets are not constantly near each other to form such a wireless sensor network.

1.2.3 Related products designed for humans

Besides solutions for pets, there are other devices close to that of our study. These products inspire us to develop the collar (e.g., physical design, bundled sensors) and the mobile application's functionalities (e.g., user interface, functions). They are the following:

- Oblu (oblu IoT, 2019). Oblu is a universal sensing platform for IoT using BLE 4.1 for communication. This development kit contains sensors such as an accelerometer, a gyroscope, and GPS, and it is a potential candidate to build our wearable device.
- Flora (FLORA, 2019). The Flora board is a small (1.75" diameter, 4.4 grams) wearable electronic platform. It is a typical representation of many Arduino compatible devices. This board only contains a microcontroller and programming circuits, without using any batteries and sensors. Flora is also a possible candidate for building a lightweight smart collar.

- GENEActiv Wireless (Activinsights, 2019). This company offers four variations of bracelets (wristband) for humans. Depending on the variant, it includes an accelerometer, body temperature sensor, skin temperature sensor and a gyroscope, only the wireless version is equipped with wireless data transmission via a Bluetooth interface to transfer data. The built-in three-axis gyroscope allows readings up to 1 kHz. The battery life ranges from 8 hours (wireless version) to 45 days (original version).
- MbientLab (mbientlab, 2019). This is a smart sensor module for motion tracking and contains an accelerometer, gyroscope, pressure, thermometer, and special mathematical preprocessor to combine data from each sensor integrated on the board. This device can be paired with a BLE gateway hub to transmit data to the cloud.
- Microsoft Band (Microsoft, 2019): Microsoft's Band is a smart wristband device that is used for human fitness and activity monitoring. This wristband contains an accelerometer with sampling frequencies up to 62 Hz. The product is discontinued and out of production now. The software support is also expired in 2019.
- Fitbit (Fitbit, 2019) is another human activities' tracker. It is small (e.g. wrist or pocket size) and contains various sensors to track steps and estimate energy consumption, sleeping time, and daily activities. The battery life is around 3 days and the device can store accumulated data of the time span of 1 week. The Fitbit device transfers data wirelessly to a base station which is also a charging dock. This device is a great example of popular wearable devices in the market.
- Xiaomi MiBand (Mi, 2019). This is a fitness tracker for humans. It contains an accelerometer and notification LEDs. With the original manufacturer's firmware, this device only provides aggregated data. The battery can last up to 30 days.

1.3 Behavior analysis with wearable devices

1.3.1 Human activity recognition

Thanks to the popularity of human wearable devices, behavior analysis using wearable devices' data attracts a lot of attention recently (Lane, Bhattacharya, Georgiev, Forlivesi & Kawsar, 2015; Tolba, Said & Al-Makhadmeh, 2019; Jalal, Quaid & Hasan, 2018; Lee, Chong & Lee, 2016).

For human activities, (Papagiannaki et al., 2019) investigates methodologies to recognize movement behaviors of senior people. First, the data are collected from a 3D accelerometer, 3D gyroscope, and 3D magnetometer, which are associated with activities such as standing, walking, going upstairs/downstairs, or lying down. The authors then perform behavior classification using standard machine learning as well as deep learning techniques. It shows that deep network outperforms conventional machine learning algorithms. It is noted that in this study the analysis part is done offline on a separate computer. Due to the complexity of data and algorithms, the experiments were operated on an Intel(R) Xeon(R) @ 3.70GHz processor with 8GB RAM and a GPU (Titan Xp 12GB VRAM). Although the results show positive of deep network method, it is not feasible to replicate on low-power devices such as mobile phones or limited-power hardware systems (e.g., Raspberry).

In the study of (Bayat, Pomplun & Tran, 2014), the authors perform daily human activity recognition (such as dancing, running, slow walk, etc.) using accelerometer data from smartphones. They employ several machine methods like logistic, multiple perceptions, random forest, and logit boost. The results (F-measures) are high (>90%) for these algorithms indicates good accuracy of activity recognition.

There are also other studies that utilize accelerometer data to detect human activities by using multiple standard machine algorithms (e.g., SVM, K-nn) (Lester, Choudhury & Borriello, 2006; Mannini & Sabatini, 2010; Foerster & Fahrenberg, 2000)) with high accuracy (>74%).

The work of (Zhang & Zhang, 2019) proposes another method to recognize human activity based on motion sensor data. Although the authors use U-net (Ronneberger, Fischer & Brox, 2015a) - a deep learning architecture - for this task on an existed data-set. The results show that their method is better than conventional machine learning algorithms (F-sw score is greater than 0.88 in U-net and greater than 0.78 for classical machine learning algorithms).

In general, despite the differences in algorithms and methodologies between studies, it shows that conventional machine learning is one of the methods for human behavior recognition applications.

1.3.2 Pet and animal activity recognition

(Haladjian *et al.*, 2017) propose a system to track behavior for free-roaming pigs. The data are from a 6-axis Inertial Measurement Unit (IMU) and a BLE module. The device is affixed to the body of pigs when the pigs are practicing their daily life activities. The action of walking, eating, and resting are recorded and labeled via video with its sensor data for training purposes. Later, that sensor data are classified by using SVM, kNN, and linear discriminant. The results show the accuracy of 95.8% with a low misprediction rate (precision: 75,4% and recall: 86,6%). It is an indicator that we could apply a similar approach to this research.

In (Ladha *et al.*, 2013b), they develop a collar-based motion sensor platform for dogs' activity recognition with an overall recognition accuracy of 68.6% via KNN algorithm.

Similarly, in the works of (Kumpulainen *et al.*, 2018), they used a 3D accelerometer movement sensor placed on the dog collar to evaluate and classify seven activities (such as sitting, sniffing, walking, etc.) of 24 dogs. The test set is annotated with the help of a video camera. The data are trained by Linear and Quadratic Discriminant Analysis with the highest overall classification accuracy for the seven behaviors was 83%.

Even for some complex movement behavior such as scratching and eating, by using standard machine learning, they can be detected (Kumpulainen *et al.*, 2018).

(Haladjian *et al.*, 2018) proposes a wearable device affixed to one leg of a cow for gait anomaly detection. The accelerometer sensor data from devices are classified via SVM to know when the cow has problems in standing postures.

Another approach recently is to use deep neural network (i.e. deep learning or multiple perception network) for data analysis. In the work of (Liu, Xuan, Hussain & Chong, 2019), the authors built a video camera monitoring system to track the behavior of pigs. They apply deep learning techniques on the recorded to recognize the pig activities.

In summary, it shows that besides using a computing-intensive machine learning method (i.e., deep learning), it is common to use standard machine learning algorithms, and acquires high accuracy.

1.4 Current research limitations and challenges

There are limitations in aforementioned pieces of research related to building smart wearable device systems for animals and how collected sensor data are analyzed.

At first, some of them are not portable wearable devices and need a constant source of power such as (Mookkie, 2019; Connect, 2020; Sec *et al.*, 2018b; Liu *et al.*, 2019) or too heavy for domestic animals (Guo *et al.*, 2006). At the same time, many of them are designed to be a single purpose function such as a feeder, and a smart bowl (Mookkie, 2019; Connect, 2020) or a GPS tracking device (Scollar, 2019). They create a closed system and do not provide a standard way to connect to more devices or services. There is one exception that is (Guo *et al.*, 2006), which can be extended by adding the same device on several cattle. However, this system requires animals to be close to each other in order to form a mesh network. This is not practical for domestic pets (e.g., cat or dog), which has relatively small numbers in one household and are not constantly staying at a specific location.

In terms of software application, many of them have integrated-wireless communication but are lacking pet behavior analysis. These shortages can be seen in multiple systems like (Mookkie, 2019; Sec et al., 2018a; Weiss *et al.*, 2013).

Moreover, many systems help to monitor the well-being of dogs or cats and they can do that in a short period of time because of lacking a large permanent sensor data storage system (Weiss *et al.*, 2013; Guo *et al.*, 2006; Connect, 2020).

Multiple products provide a mobile application to monitor and control their devices like Mookkie (2019). Moreover, many projects do not provide such utilities like (Guo *et al.*, 2006; Pongrácz Rossi et al., 2016; Hu, Silver & Trude, 2008). This is not crucial for the system to work but substantially reduces their usability because of the popularity of smart mobile phones (66% of Canadian possess one smartphone according to (CENTER, 2020)).

A system that can overcome those shortcomings must solve challenges in both hardware and software. In terms of hardware, it must be light weight enough to affix to domestic pets. The device must also equip with sensors to monitor the surrounding environment and the physical situation of pets. These sensors could be an ambient sensor to get air temperature, a sound sensor to listen to the surrounding environment, an accelerometer sensor to track the motion in 3D, and so on. Many other conditions about the physical requirements of a device such as battery life long, outside shape, size of the device, communication protocols, and durability of a device's case should be optimized for usages. In terms of software, the system should provide an advanced data analysis application to extract the well-being data of pets. In particular, it should be able to recognize the pet behaviors and provide the pet activity summary overtime. In addition, it should include one mobile application to interact with the system and, that mobile application must be developed for one of the most widely used operating systems. For details, Table 1.1 summarizes the limitations of multiple pieces of research and commercial products.

Table 1.1 Feature comparison between proposed and existed systems

	Wireless controlling	**Behavior recognition**	**Large data storage**	**Open architecture system**	**Wearable**	**Mobile software**	**For domestic pet**	**References**
Mookie	Yes	No	N/A		No	Yes	Yes	(Mookkie, 2019)
Feeder Connect	Yes	No	No	No	No	Yes	yes	(Connect, 2020)
PetPace	Yes	Yes	Yes	No	Yes	Yes	Yes	(Petpace, 2019)
Fitbark	Yes	Yes	N/A	No	Yes	Yes	Yes	(FitBark, 2019)
Scollar	Yes	No	N/A	No	Yes	Yes	Yes	(Scollar, 2019)
Petbuddy	Yes	No	No	No	Yes	No	Yes	(Ahn *et al.*, 2016)
WagTag	N/A	N/A	N/A	No	Yes	N/A	Yes	(Weiss *et al.*, 2013)
David Sec et. al.	Yes	No	N/A	No	Yes	No	No	(Sec *et al.*, 2018a)
Y. Gou et. al.	Yes	No	No	No	Yes	No	Yes	(Guo *et al.*, 2006)
Dog using skype	Yes	No	No	No	No	No	Yes	(Pongrácz Rossi *et al.*, 2016)
LonelyDog@Home	Yes	No	No	No	No	No	Yes	(Hu *et al.*, 2008)
Proposed system	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	

1.5 Proposed system

Although smart toy systems for pets with the targeted specifications and objectives start to appear on the market, they are often mono-function stand-alone devices such as a health monitor, an activity tracker, or a food dispenser. Many are heavy and require strong power supply sources such as big battery packs or power lines. Hence, users still end up having multiple systems to handle, which is undesired hindering gaining interest in such systems among pet owners.

At the same time, several systems do not have the data analysis applications. This software is important to monitor the well-being of pets because it can help to detect abnormal pet activities. Thus, the owner might anticipate the appropriate decisions such as medical intervention.

The proposed study aims to contribute to overcoming these issues and offer multi-use smart products for pets (owners) and a behavior recognition application. In detail, we propose architecture designs for smart toy system for pets consisting of a collar, toys, other connected devices and applications to analyse pet behaviours based on sensor data.

The proposed system tries to explore possibilities to fill all those gaps (indicated by “No“ features in Table 1.1). The proposed collar is designed to contain sensors for ambient , magnetic field, and acceleration. However, the collar prototype has only one acceleration sensor at this research stage. This prototype is developed by BeOneBreed a local industrial partner - who possesses extensive expertise in making toys for pets. This prototype system includes this collar attached to the pets, a mobile application and cloud applications.

1.6 Chapter summary

In this chapter, we iterated over multiple related works and technologies to build a system of smart toys for pets. By investigating products in the market and research in academia, we pointed out the gaps and limitations in current systems (e.g., not wearable).

After that, we discussed current data analysis methods commonly used in pet behavior prediction. It is found that conventional machine algorithms still prove to be a suitable and effective way to forecast animal behaviors.

Finally, we proposed a smart toy system for pets with integrated pet behavior analysis applications. This system consists of a collar, a mobile application, and data analysis components.

CHAPTER 2

BACKGROUND

The proposed smart toy system for pets should include a wearable device affixed on the animal, a wireless control application on the mobile phone, and possible surrounding connected devices. In order to develop such systems, there are several related tools involved such as communication technologies (Bluetooth), application development environment, data storage approaches and machine learning analysis methods.

2.1 Internet of Things

In (Van Kranenburg, 2008), Internet of Things (IoT) is defined as “a dynamic global network infrastructure with self-configuring capabilities based on standard and inter-operable communication protocols“ and where physical and virtual ”Things” are “seamlessly integrated into the information network“. Generally, any devices that have sensors or embedded software and can connect with other devices can be called an IoT device. The connection uses wireless technologies (e.g., WiFi, BLE). Figure 2.1 shows the interconnection of an IoT device inside its network.

2.2 Wireless communication

2.2.1 WiFi

WiFi (802.11) is a standard for wireless communication between devices such as computers, and smartphones. It operates in 2.4 and 5 GHz frequency bands. WiFi standard provides the connection to the nearest access point and is able to cover a small space such as within a room or a few rooms. However, it is not designed for low-power devices. Therefore, other standards are made for this purpose such as Bluetooth Low Energy (BLE). Table 2.1 is a comparison between these two wireless standards.

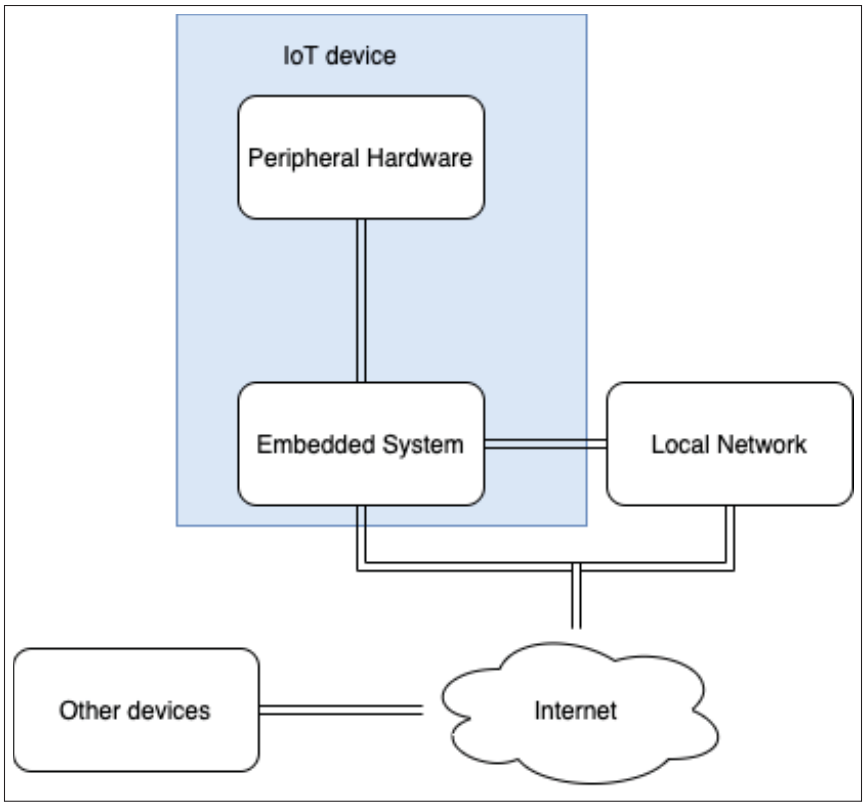

Figure 2.1 An example of an IoT device connected to its networks

WiFi can be used to expose the connection between mobile phones or IoT hubs to the Internet while BLE is for communication between smart connected devices or between mobile phones and smart devices.

2.2.2 Bluetooth Low Energy

Bluetooth Low Energy (BLE or IEEE 802.15.1) is a part of the Bluetooth 4.0 Core specification. It is a standard for low cost, bandwidth, power, and complexity. It is used for secured connecting and transferring data between devices in a short-range (usually within 10 meters).

A BLE device can communicate with the world in two ways:

- *Broadcasting*: this is a connect-less communication (one-way transmitting). The device sends out data to any scanning devices or receivers within its listening range.

- *Connection*: it is for transmitting data in two directions. A connection is a persistent and timely exchange of data between two devices.

Table 2.1 WIFI and BLE comparison

Characteristics	**WiFi**	**BLE**
Nominal range	150m	10m
Frequency band	2.4, 5 GHz	2.4 GHz
Max data rate	1Gbps	2Mbps
Latency (ms)	150	200

Table 2.2 Operation power consumption of WiFi and BLE

Power consumption	**high (WiFi)**	**low (BLE)**
Sleeping mode	10μW	8μW
Transmit power	350μW	26.5μW
Receive power	90μW	28.5μW
Average power for 10 messages per day	500μW	50μW

In terms of power consumption, (Putra, Pratama, Lazovik & Aiello, 2017) shows that average power consumption of WiFi is ten times bigger than that of BLE for 10 messages per day (Table 2.2).

Similarly, (Lindemann, Schnor, Sohre & Vogel, 2016) measured device runtime and found out that BLE devices can significantly run longer compared to WiFi devices (e.g., 80 hours for BLE and 151 hours for WiFi).

2.3 Mobile application development

We selected iOS as a supporting operating system because of its popularity. Along with Google Android, Apple iOS is one of the popular mobile operating systems. In 2016, Apple announced that it had one billion devices worldwide [1].

[1] http://images.apple.com/pr/pdf/q1fy16supplementalmaterial.pdf

In order to develop a mobile application for iOS, it is required to use a MacOS-running computer (e.g., Macbook Pro), Xcode (see Figure 2.2) — a development environment, and a subscribed Apple developer account. For testing purposes, it is also needed to have at least one device such as an iPhone or iPad because the BLE functions are not supported on an emulator and can only run on a physical device.

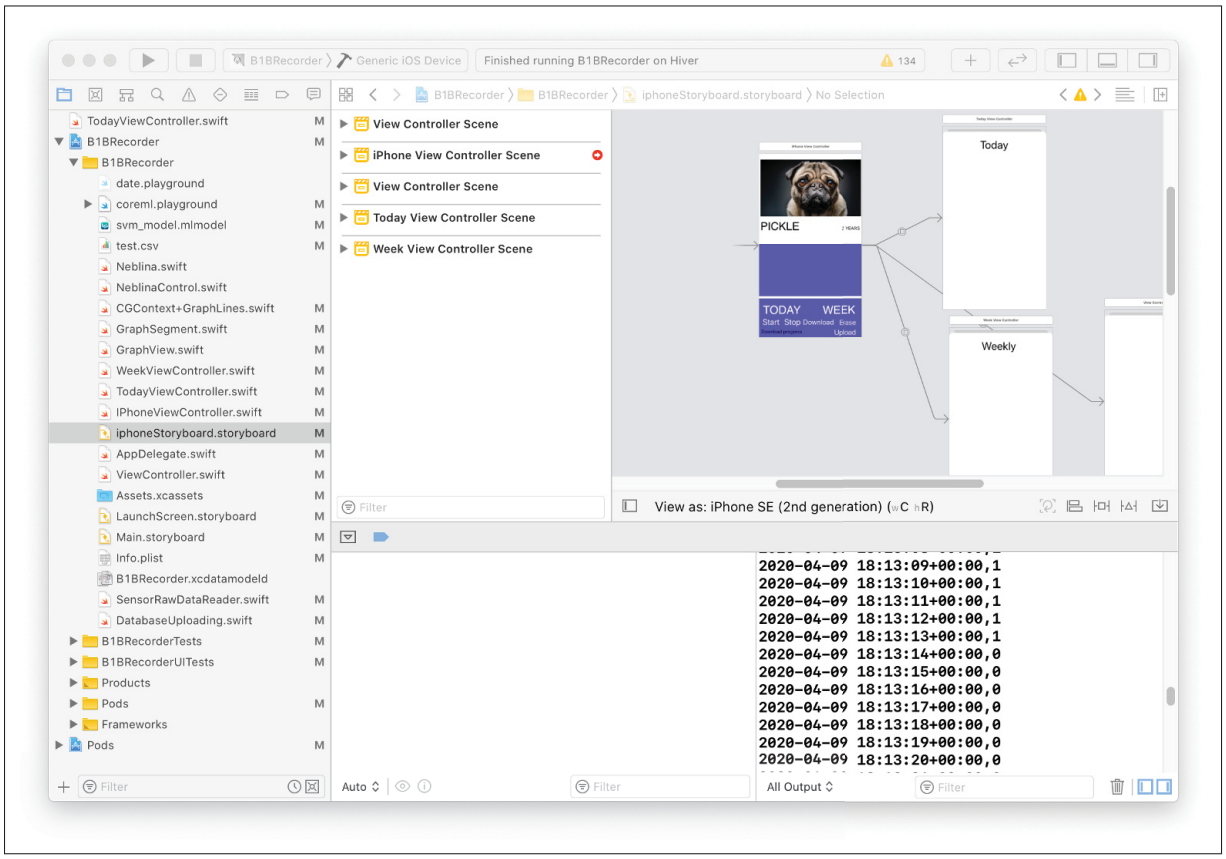

Figure 2.2 User interface of Xcode during the mobile application developing time

Particularly, we use Swift as the programming language because it is better compatible with existing libraries. Furthermore, we also utilize a recently introduced technology called Core ML (Apple, 2020) (Figure 2.3). CoreML helps to integrate existing machine learning models (called Core ML model) into the applications.

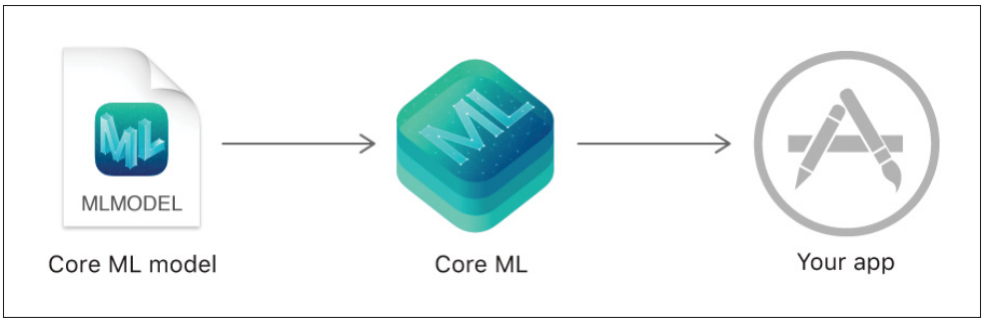

Figure 2.3 Overview of Core ML usage
Taken from Apple (2020)

2.4 Machine learning analysis methodology for wearable devices

In this section, we are going to introduce general knowledge about machine learning and their taxonomy. We also argue which approaches are suitable for studying data collected from sensors of wearable devices.

Machine learning is a sub-field of computer science. It provides the machine with the ability to learn rules from given examples without explicit instructions. Machine learning originates from pattern recognition and computational theory. Here, we are going to discuss some core concepts of machine learning as well as examples of machine learning in data analysis for wearable devices.

A machine learning algorithm takes sets of examples as input called *training data*. Based on the differences of training characters, machine learning algorithms can be divided into *supervised, unsupervised*, *reinforcement,* and *deep learning* (Barber, 2012; Bishop, 2006; Robert, 2014; Goodfellow, Bengio & Courville, 2016) (Figure 2.4).

In supervised learning, the training data consists of the input vectors associated with their output values, which is called *label*. For example, the input vectors could be accelerometer sensor data and the labels are their corresponding activities. The objective of supervised learning is to learn how to predict the outputs based on the input values. In other words, the algorithm has a

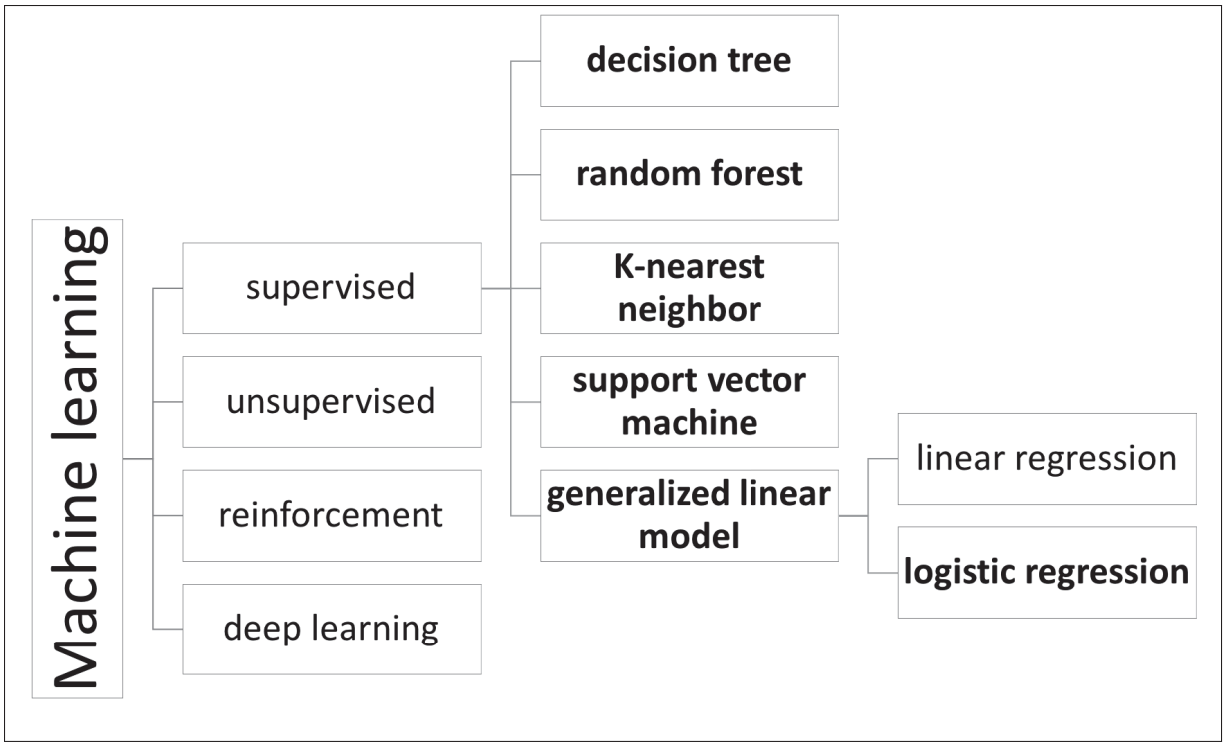

Figure 2.4 Simplified taxonomy of machine learning algorithms based on training data

clear goal to go. An application that has a finite set of labels is called *classification*. If it has an infinite set of labels (i.e., continuous values), that task is called *regression*.

Decision tree is a popular supervised algorithm for classification data. However, it is extremely sensitive to perturbations in data. A small change can result in a considerable tree. It is easy to be over-fitting and has the problem of out-of-sample prediction (hard to predict a new unseen sample). Because of their limits, an enhanced version - ensemble learning version called random forest is used for sensor data (Bayat *et al.*, 2014; Caruana & Niculescu-Mizil, 2006; Pal, 2005; Chaudhuri et al., 2018)

Support vector machine is another supervised algorithm that can be used to do classification on wearable sensor data (Haladjian *et al.*, 2017; Kumpulainen *et al.*, 2018) and shows good results. K-nearest neighbor is used in several studies to predict the behavior of animals such as (Haladjian *et al.*, 2017; Ladha *et al.*, 2013b).

Linear regression is also a supervised learning algorithm for continuous values, which is not suitable for classification tasks. Another algorithm in the same family of general linear model - logistic regression is a perfect fit for binary classification tasks in this research.

On the other hand, unsupervised learning does not have a well-defined objective. It does not have labels for the training and tries to extract insight information from vague pre-defined objectives (for example, it groups data into *n* group without knowledge about those group - *n* is usually given by experimenters). In this group, there are algorithms such as principal component analysis (PCA), neural network, or Gaussian mixture model. One of the major applications of unsupervised learning is to identify the likeness between one data and other groups of data. This process is called *clustering*. In our research, the training data is labeled so that we do not apply unsupervised learning.

Reinforcement learning follows a different approach. Its goal is to maximize the payoff of the algorithm given what actions or sequences of actions to be taken. It is also not required labeled data for model learning. This machine learning method usually uses in an interactive system where received feedback is used to determine the next system action (Mahadevan & Connell, 1992; Kaelbling, Littman & Moore, 1996). This problem is not closed to our research because there is no ‘feedback‘ to change the learning outcomes.

Deep learning is a different approach using multiple layers (i.e., deep) neural network. It can be used both in supervised and unsupervised tasks.Deep learning algorithms usually uses for unstructured data such as text, audio, and, images. Besides, deep learning requires a substantially huge amount of training data in order to generalize results. In this research, the training data is a relatively small data collection within a few hours (more details are in Section 4.4). Because of that, we do not use deep learning for prediction. It might be considered in the future when the data is much more collected.

In the next sections, we are going to present logistic regression, kNN, SVM, and Random Forest classification algorithms. These algorithms are selected because of their previous successful

application in classifying sensor data of wearable devices specialized for pets (more details in Section 1.3.2).

2.4.1 Logistic regression

Traditionally, logistic regression (Hilbe, 2009) is a supervised machine learning algorithm for binary classification. It has this form

$$p(y = 1|x; w) = \sigma(w^T x) \tag{2.1}$$

where σ is the logistic *sigmoid* function given as,

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

Given an unseen data point, logistic regression emits the probability that the new data belongs to class 1. The results of logistic regression can be interpreted as the likelihood of the conditional distribution $p(y|x)$ where y is the class of x. For advanced information, (Yang & Loog, 2018) provides an updated benchmark of learning capabilities of Logistic Regression.

2.4.2 K-Nearest Neighbors

K-Nearest Neighbors (KNN) is a supervised algorithm. In KNN, the goal is to classify the new unseen input point by looking at K given points in the training set that are closest to it in the feature space. In order to find the k-nearest points, it uses distance, such as Euclidean, or Hamming distance. Let denote that new input vector (data point) by x, its K nearest neighbor by $N_k(x)$. The predicted class of x is denoted by y, class by c. $I(t = c)$ is the indicator function, which value is 1 if t is in class c, and 0 otherwise. The variable $p(t = c|x, K)$ is the probability that point x is in class c given its K neighbors. The formula of classification is

$$y = \arg\max_c p(t = c|x, K) \tag{2.3}$$

where y is the class of x and $p(t = c|x, K)$ is denoted by

$$p(t = c|x, K) = \frac{1}{K} \sum I(t = c) \tag{2.4}$$

One problem with KNN is that it must store and process all training data for every new input point x. Therefore, it is not suitable if the training set is large or memory on a device is relatively small. To overcome this disadvantage, many enhanced versions of KNN are proposed such as works of (Jagadish, Ooi, Tan, Yu & Zhang, 2005; Do, Douzal-Chouakria, Marié & Rombaut, 2015).

2.4.3 Support vector machine

Classical support vector machine (SVM) is a binary classifier that aims to find a separated hyperplane between both classes of the training set with the maximum margin. This hyperplane can be visualized as an example in Figure 2.5). The label (i.e., class) of a new unseen data point is decided by which side of the hyperplane it falls into (Cortes & Vapnik, 1995). When the data is not linearly separable, it is transformed into a higher dimension space which in turn becomes linear separable. This transformation function is called *kernel*. There are several common kernels such as Gaussian-kernel (Karatzoglou, Smola, Hornik & Zeileis, 2004; Varewyck & Martens, 2010)), and polynomial kernel (Chen, Yuan, Hua, Zheng & Wang, 2015; Yaman & Pelecanos, 2013). SVM can be used for regression and classification (Ben-Hur, Horn, Siegelmann & Vapnik, 2001, 2000; Xiao & Eckert, 2013) tasks.

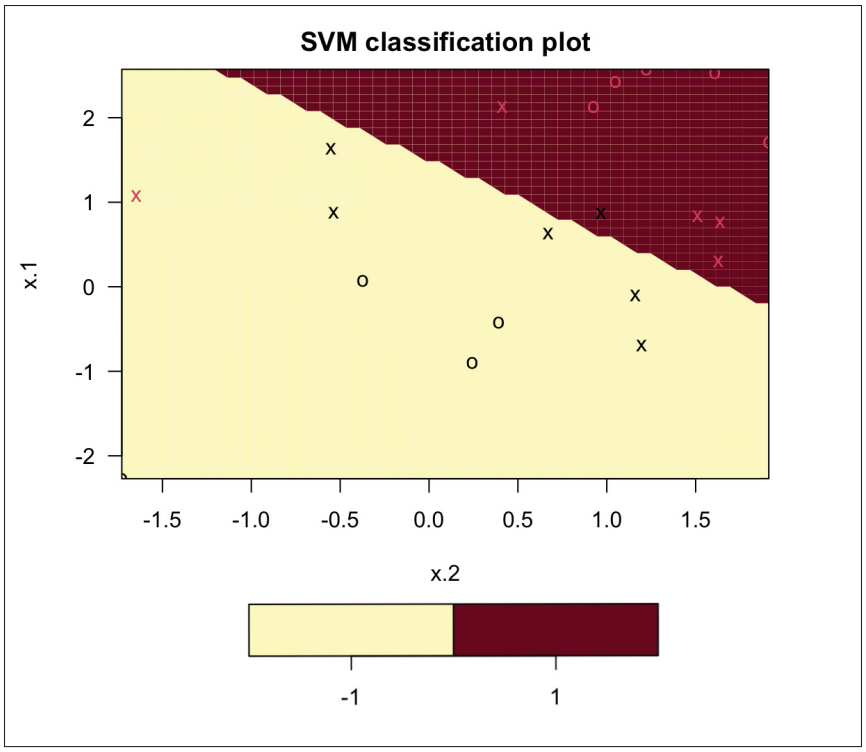

Figure 2.5 Using SVM to classify class -1 and 1

2.4.4 Random Forest classifier

Instead of using a single machine-learning algorithm to model the training data, multiple algorithms are applied at the same time to produce a better prediction. This technique is called *ensemble learning*. Random forest classifier is an ensemble machine learning algorithm which uses multiple decision trees to model the data. The prediction results are decided by major-voting (e.g., mean or mode prediction) of all individual trees. A simple visualization of this process can be seen in Figure 2.6. Particularly, Figure 2.7 is an example of a single decision tree inside a random forest to classify ‘iris‘ data-set. Each rectangle box contains information which is used for classification.

Random forest classifier is a fast and scalable algorithm to handle large datasets. It works well even without standardizing input. Its accuracy is high accuracy but hard for a human to interpret the results (Caruana & Niculescu-Mizil, 2006).

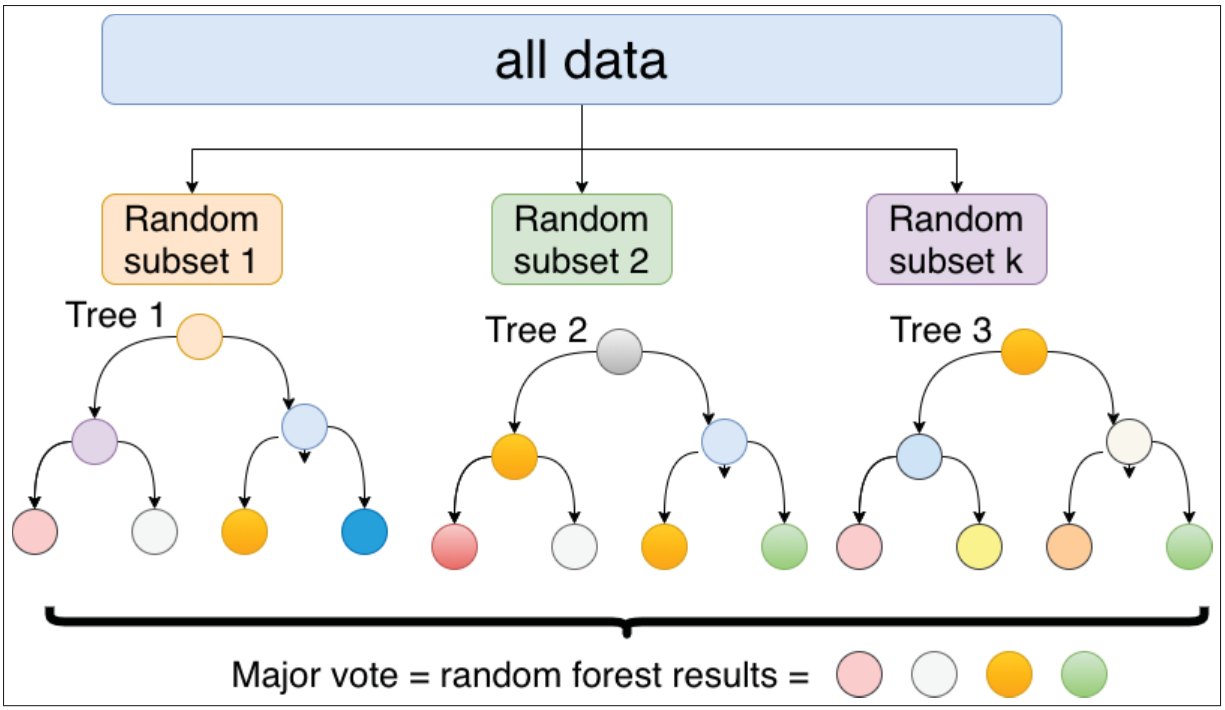

Figure 2.6 Random forest classifier visualization

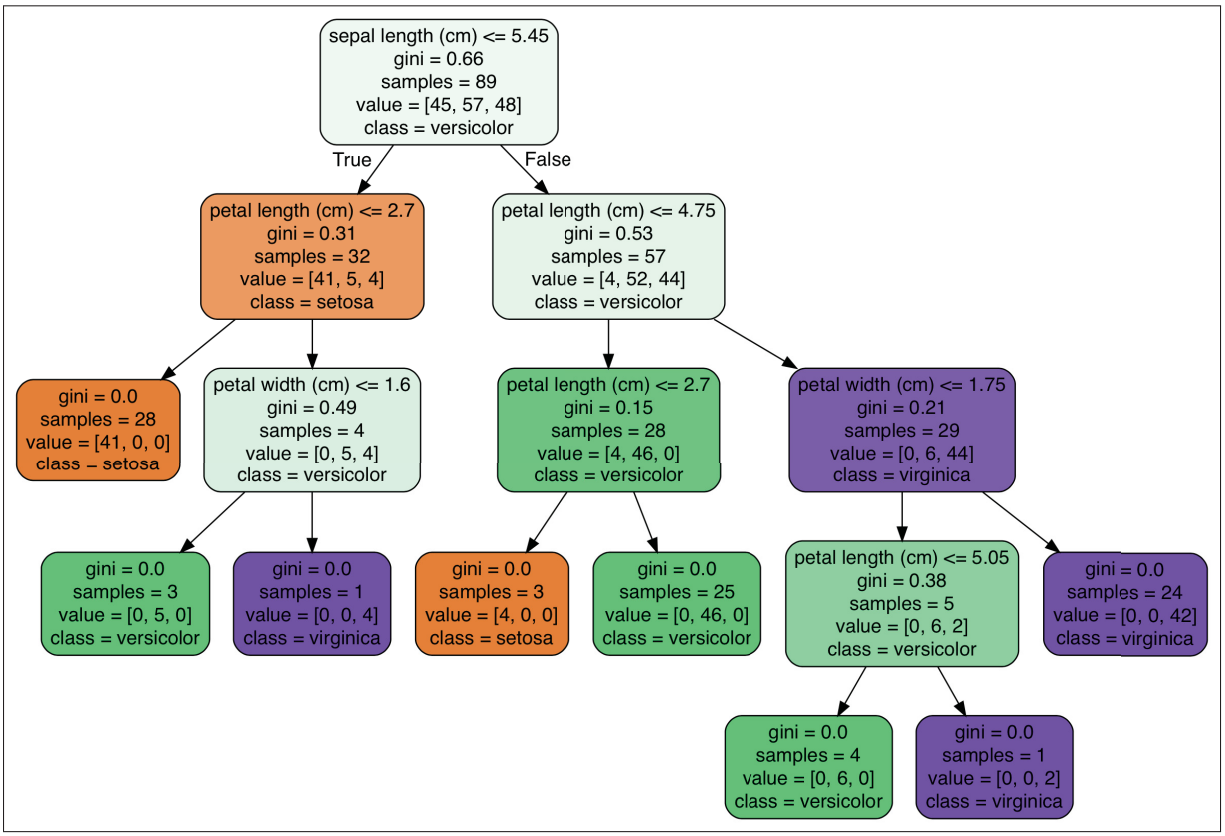

Figure 2.7 A decision tree to classify iris flower dataset

2.5 Chapter summary

In this chapter, we iterated the backgrounds of IoT devices, and popular wireless communication technologies to provide the general picture of and IoT connected systems.

After that, we discussed the mobile application development processes and its related tools. We then introduced the machine learning analysis approach and its taxonomy. Finally, we presented four (4) popular machine learning algorithms that can be used to classify sensor data.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter discusses the methodology to build the proposed smart toy system for pets. Firstly, we discuss the use cases of a smart toy system to provide an overview of the system components. Secondly, we investigate on arguments to design of each component. Finally, we propose possible architectures for this kind of system.

3.2 Use cases of the system

We hypothesize that the system has the following five components (see Figure 3.1):

- a collar affixed to the pet
- a mobile application running in the owner's phone
- the other connected smart toys
- the hub to connect all devices
- a cloud part for data storage and analysis

These components can communicate to each other via wireless technologies (e.g., WiFi, BLE). The arguments of having these components are based on our presumed system usages. We assume that the proposed smart toy system for pet can work in multiple situations according to the relative proximity between the pet and its owner. The system has to be not only helpful when the owner and his pets are inside a room but must also be useful when they are outside of their living spaces.

To better the functionalities of the system, we are going to investigate certain connected devices and multiple usage scenarios. First, we are going to explore several devices (smart toys, smart sensors and hub) and understand how they can be integrated into the system. Later, we are going to examine 4 system use cases in Section 3.2.1, 3.2.2, 3.2.3 and 3.2.4. In each use case, we

discuss how the system works in different situations and roles of each of its component. By doing this, we can provide a clearer understanding on how the system can help the pet and its owner.

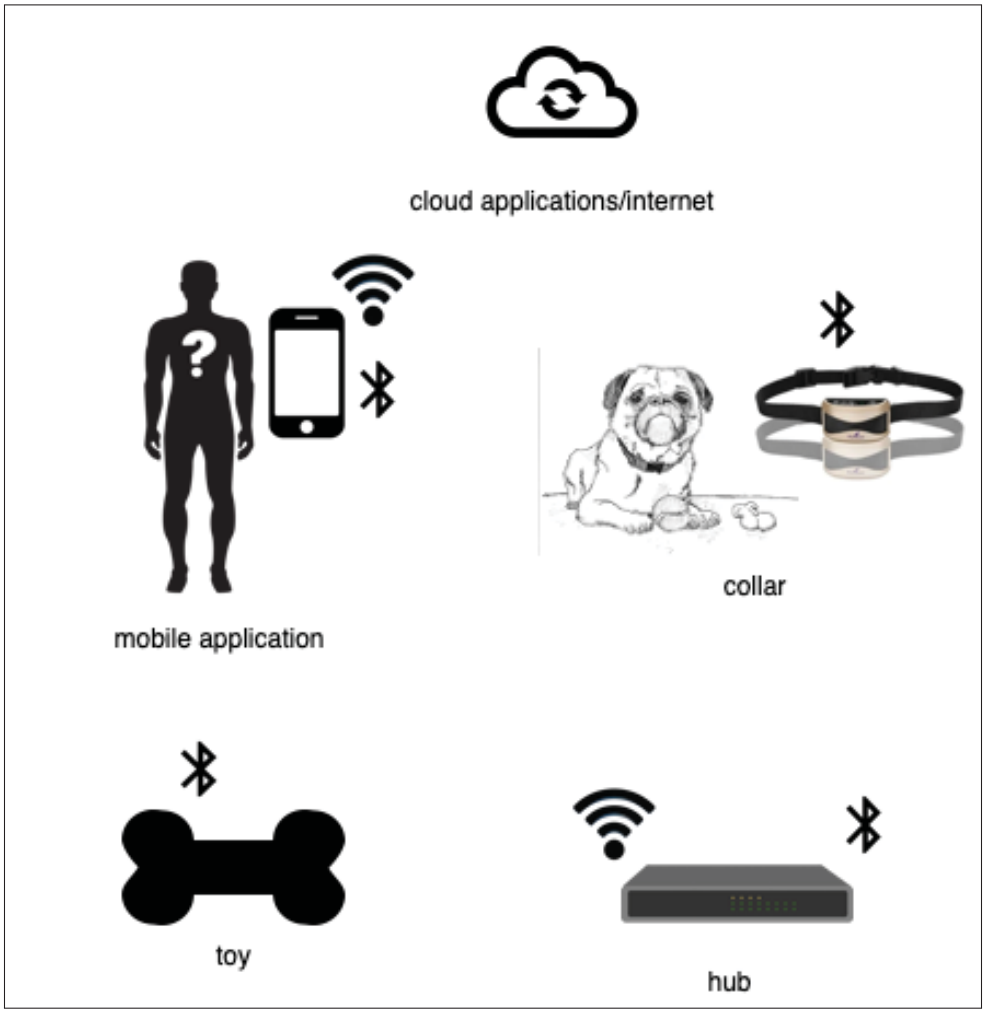

Figure 3.1 List of components of the proposed smart system for pets

It is important that the collar works inside a system with several types of smart toys or connected devices. Because of that, we propose some possible usages for such devices. In this use case, we take a smart toy, a smart sensor, and a hub to explain the scenarios.

- **Smart toys** - The smart toys are important for the extensibility of the system. It is expected that several future toys can be added into the system. In this paragraph, we propose one example: a food dispenser. When a pet wearing collar is near this food dispenser, and

depending on the settings of the owner, the collar could activate a trigger command to a toy and that toy would release the foods outside (see Figure 3.2). In order to do that, there could be a programmable function to set conditions firing the trigger command. This function could be set by the owner via the mobile application.

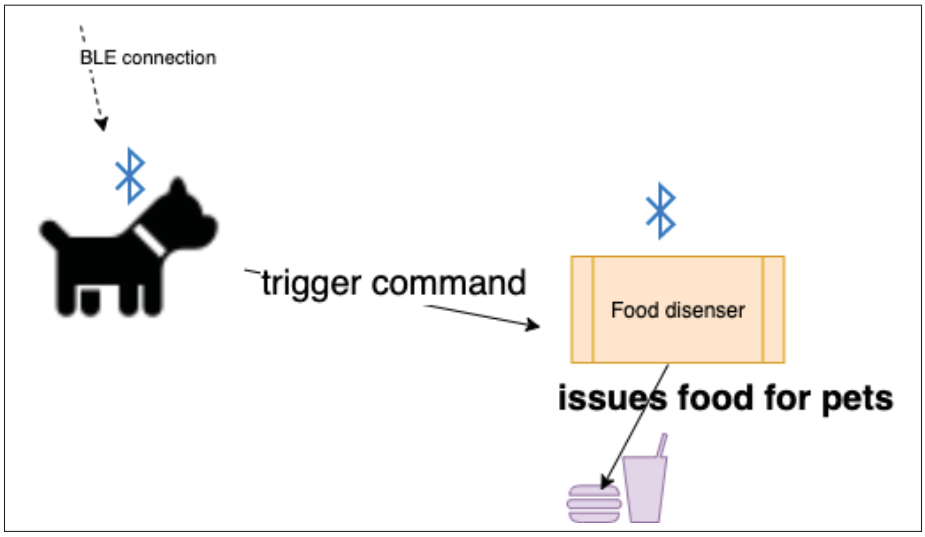

Figure 3.2 Interaction between a food dispenser and a collar

- **Smart sensor** - Another example of how to extend the system is by using a small device called *smart sensor*. This device can be attached to the door, dinner table or anywhere inside the house. For example, with the aim to prevent the pet enters the sleeping room, a detector is affixed to the door. That device is always broadcasting its identity to the surrounding environment. When the pet is inside its alert range, the collar notifies the sensor of its presence. The sensor then triggers a device (such as inaudible speaker or a lamp) to distract the pet . In this way, it is expected that the pet will not enter the bedroom. The whole operation can be seen in Figure 3.3.
- **Hub** - When the mobile phone with the application is not in connectivity range of the collar or when it is not available to use, a hub is a replacement. In terms of functionalities, this hub can control the collar, collect and perform data analysis, and synchronize sensor data with an external database over the Internet.

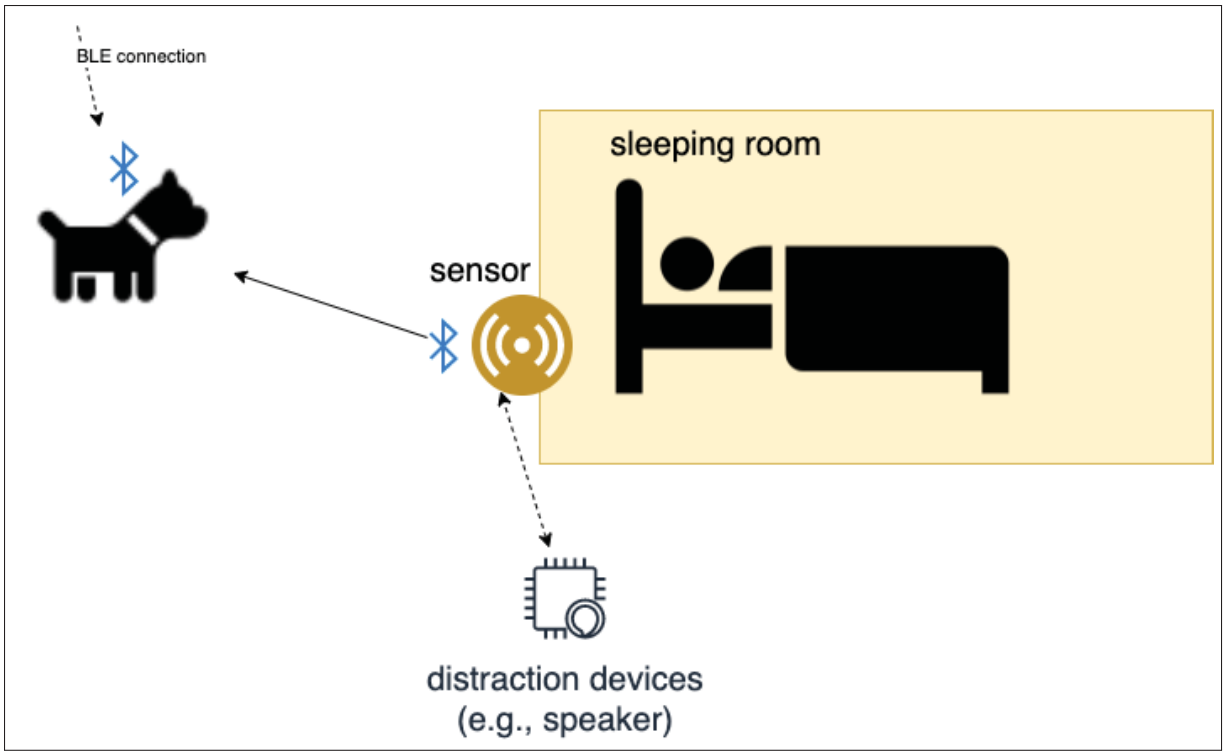

Figure 3.3 Communication between the collar and the sensor

3.2.1 Scenario 1: the mobile application within wireless connectivity range of the collar

In this use case, the owner and his pet are staying at their home. Therefore, they are relatively close to each other or can be near each other at any moment. In other words, the devices are within their connectivity range (see Figure 3.4). The user (owner) can check health and activity summary of the pet at any moment, even in a real-time manner. To do that, the mobile application connects to the collar via BLE or WiFi to get the sensor data. The data then are analyzed (inside and the phone and at an application in the cloud) and the summary is shown on the mobile screen. Based on that summary information, the owner can trigger a toy or device which could help to improve the pet well-being. For example, the owner uses the mobile phone to activate a moving toy to nudge the pet to chase it like a physical exercise. Furthermore, instead of directly triggering the toy via the mobile application, she can interact (e.g., talking) to Amazon Alexa or Google Home to do the tasks.

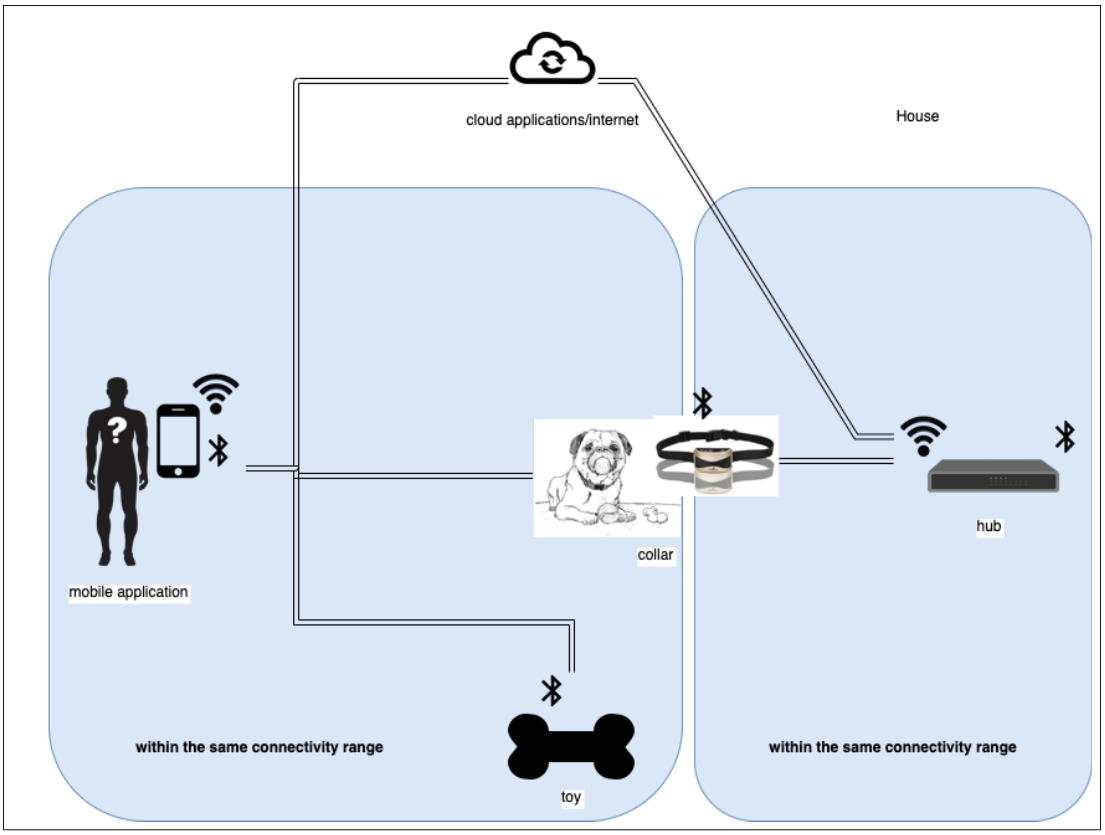

Figure 3.4 Scenario 1: the mobile application within wireless connectivity range of the collar

3.2.2 Scenario 2: the mobile application within wireless connectivity range of the collar, the hub is not

It is common that the owner and his pet are not in their house. They are in public space such as a park, an open field or a garden. This use case is popular for a dog's owner because walking or playing with the dog outside is a typical activity.

In this situation (Figure 3.5 is an example), the user has the phone with the mobile application and the pet has an affixed collar. Two devices connect to each other wirelessly to exchange data: the phone downloads sensor data from the collar and sends commands such as turning off some collar functions to save battery life. At the same time, the phone can synchronize these data with a cloud application. In the mobile application, the user can check the activity analysis at the moment of his pet and compare with previous analyses on the cloud.

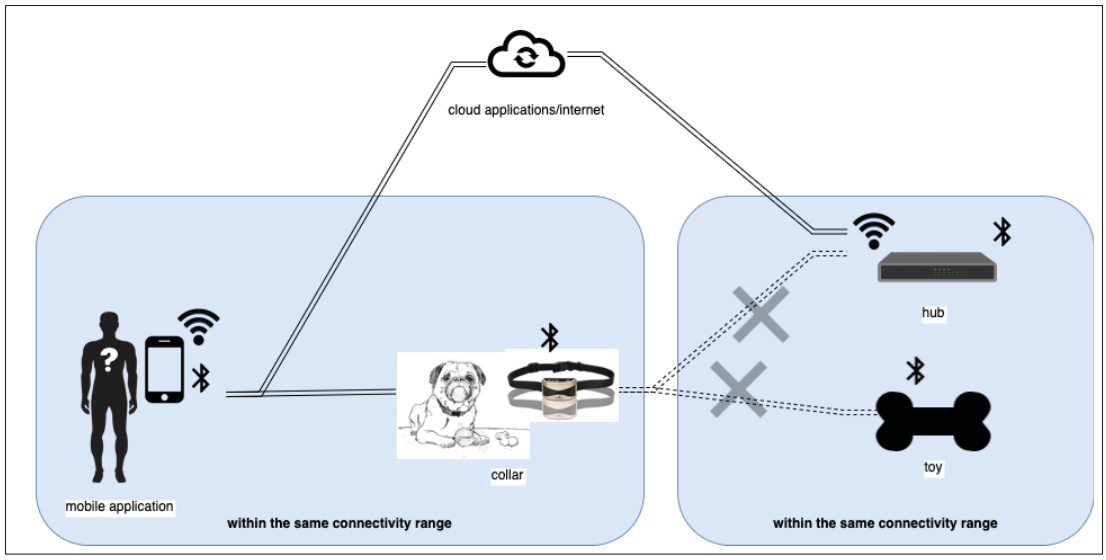

Figure 3.5 Scenario 2: the mobile application within wireless connectivity range of the collar and the hub while the toys are out of that range

3.2.3 Scenario 3: the mobile application is out of the collar connectivity range and has internet connection

In this scenario, the owner is at a workplace and the pet is left at home. Therefore, she cannot directly use the mobile application to monitor the pet behavior or trigger the collar and other connected toys. Instead, the user does that indirectly to the hub via the cloud part. The mobile application retrieve data from and send commands to the cloud , which in turn relays them to the collar. The hub is a smart device which has all of the function of the mobile application, but it is stationary inside the house and connected to the Internet. That allows it to work as a proxy to execute remote commands from the user, who is far from home. The hub also uploads collected sensor data to the cloud part. An example of this scenario can be seen Figure 3.6.

3.2.4 Scenario 4: the phone is disconnected from the system

In this use case, the owner is not at home and his mobile application is disconnected from the Internet (e.g., phone is out of energy, or having no radio signal reception). The pet is staying at home with other connected devices around. The hub now plays the role of the central command

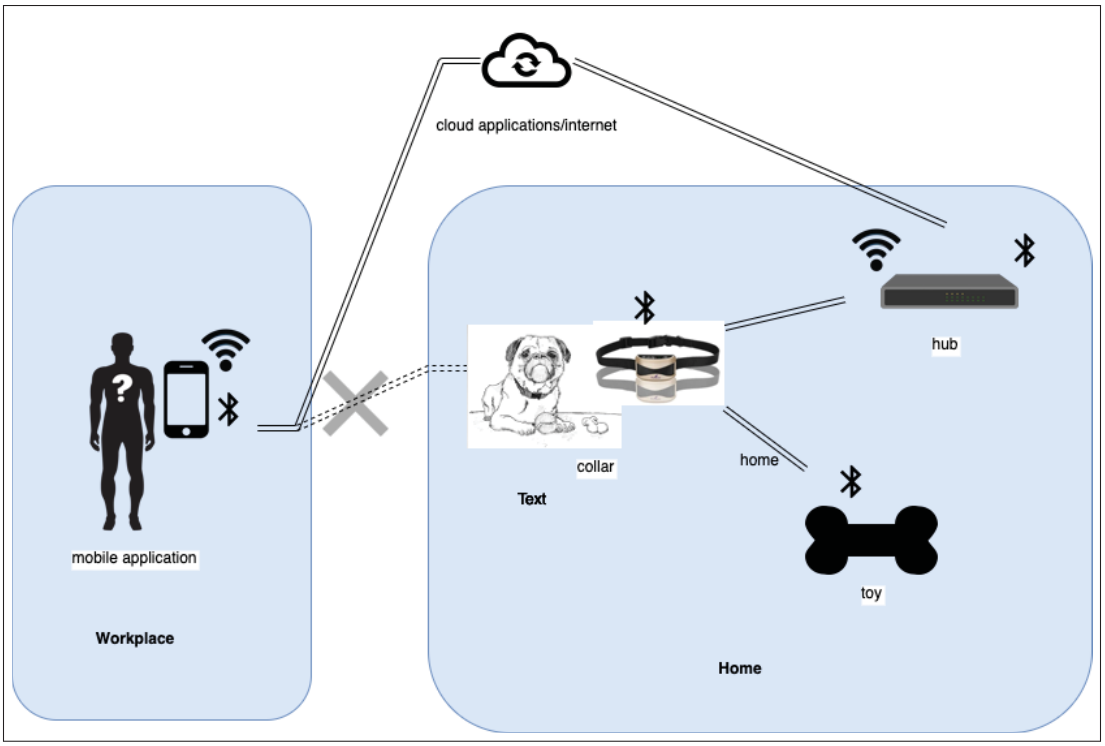

Figure 3.6 Scenario 3: the mobile application is out of the collar connectivity range and has internet connection

center. It collects sensors data from the collars, does partial data analysis and activate appropriate functions. For example, based on the analysis, it predicts that the pet is hungry, so the hub commences the food dispenser to issue foods to the pet. Also, it sends the data to the cloud for further analysis. The cloud utilizes all available data to forecast needs of the pets. At the same time, based on the user procedures which are predefined before, for instance, the owner wants the pet to lose weight by chasing the rolling toy, it will automatically send these instructions to the hub. The hub then will trigger the rolling toy when the pet is nearby. This process is done without the interaction of the pet owner. The example of this case can be seen in Figure 3.7

3.3 System components

In this section, we describe the components and their key roles in the system. They are like the following:

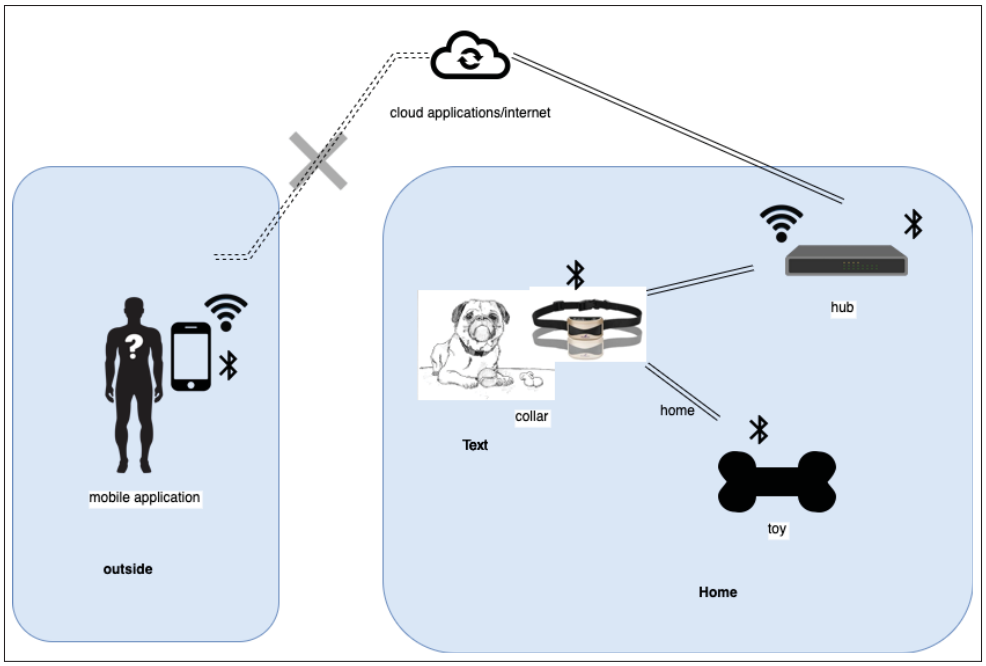

Figure 3.7 Scenario 4: the phone is disconnected from the system

1. **Collar** is used to record the pet behavior (e.g., accelerometer) or environment information (e.g., ambient temperature) via its sensors. It can also connect and exchange data with the mobile application, the hub or the connected toys.
2. **Connected devices and toys** entertain the pets (e.g., moving the ball) or provides certain specialized functions (e.g., food dispenser).
3. **Hub** communicates with toys, collar, and connected devices when they are in its connectivity range. It can also synchronize the data from the cloud part.
4. **Cloud applications** store data collecting from the hub or the mobile applications and perform data analysis part.
5. **Mobile application** controls the collars and toys; sends commands to the cloud applications.
6. **Data analysis applications** extracts insights from collected data (e.g., predict abnormally in pet activities).

3.3.1 Collar

The collar is the only device which is designed to attach to the pet body. Because of this, it is required to have suitable physical characters.

It must be light enough for the pet to wear and appropriate outer case. The weight of this device should lie below the guideline of 4-5% of body weight according to (Yonezawa, Miyaki & Rekimoto, 2009) or far less than 2% based on the study of (Valentin, Alcaidinho & Jackson, 2015). The outer shape of the collar must have enough space for its estate (e.g., board, sensor, battery, etc.) to function properly.

In terms of communication technology, the collar must use wireless method due to obviously convenient reasons. Currently, there are two of the major technologies for this: BLE and WiFi. They are considered because of the wide range of available compatible products and popularity (e.g., home network).

There are many differences between them but we focus on only two main features: coverage and power consumption. BLE has the range of 10 meters while WiFi can reach up to 150 m. Intuitively, BLE energy consumption can be up to one tenth of that of WiFi in the same condition because of its shorter range (Putra *et al.* (2017)). However, by carefully design the hardware, power consumption of WiFi-based smart devices can be lower than that of BLE (see (Thomas, Wilkie & Irvine, 2016)).

Besides, WiFi has faster data transfer rates than BLE. Therefore, if the collar prioritizes the data rate, and coverage range, WiFI should be used and power consumption must be taken into careful consideration. On the other hand, if battery life has more priority, BLE should be investigated.

For more details comparison between BLE and WiFi, one can refer to Table 2.1.

The number of sensors inside the collar has a progressive impact on its energy usage. Also, the sampling rates of a sensor also heavily impact on energy consumption (Tobola et al., 2015; Clevenger, Pfeiffer, Mackintosh, McNarry, Brønd, Arvidsson & Montoye, 2019).

Commonly, the sampling rates of sensors on human wearable device is 50Hz which is possible for the pets. However, based on the desired measurement, that sampling rates can be reduced because lower sampling rates can also detect the movement behavior.

In addition, GPS is one of the common methods to monitor the geographical location of the pet. However, a typical GPS module consumes energy from 143 to 166 mW (Carroll, Heiser et al., 2010) in the continuous navigation model, which could deplete a mobile phone's battery in around six hours. We can estimate the collar can last less than one hour if this GPS module is operated in the same condition.

Table 3.1 summarizes the technical choices for the collar.

Table 3.1 Technical decision criteria for the collar

	Pros	**Cons**
WiFi	has fast transfer rate (1Gbs), wider range (150m)	needs more energy (e.g., 350μW in sleeping mode)
BLE	consumes less energy (e.g., 26.5μW when sleep)	has smaller coverage area (10m), slow data rate (2Mbps at max)
Add more sensors	gets more vital data (e.g., accelerometer, temperature)	consumes more energy (4μA for one accelerometer sensor (Electronics, 2020))
Add GPS module	provides more accurate positions (accuracy of GPS is up to 5 meters)	has more energy consumption (143 - 166μW)
Higher sampling rates	collects precise data (with rating 100Hz, sensors collect 100 data sample each second)	consumes more energy (accelerometer consuming 210 mAh at 50Hz, consuming 294 mAh at 800Hz (Ma, Qiao, Lee & Fallon, 2013))
Lower sampling rates	conserves battery	can miss some data points
Bigger dimension	has more space for battery and hardware	is heavier, might be too heavy for pets (Panasonic CG-320: capacity 13mAh, diameter of 3.5mm and a weight of 0.6g (Telecom, 2020))
Smaller physical capacity	is convenient to affix on pets	has less space for other hardware

3.3.2 Connected toys

The toys should not have physical dimension limits. Therefore, they can operate by both wall power source or having space for large battery packs. For example, a movable toy (e.g., roller) should run on the battery while a food dispenser can be plugged its charger to the wall.

However, they must use the same communication technologies that are used on the collar or the hub. If the toys use WiFi communication methods, it would require a bigger source of energy or need to be charged more frequently but will have faster and better data transfer range. It also could send data directly to the cloud part in the Internet. If BLE is selected, their batteries can last longer but require the supporting of the hub or mobile phone for data exchanging.

Technical options for the toys (or connected devices) are summarized in Table 3.2. For comparison between WiFi and BLE, one can refer to Table 2.1.

Table 3.2 Technical decision criteria for the toys

	Pros	**Cons**
Power by battery	mobility	limits in operation time (Adafruit feather board - www.adafruit.com - has battery of 333µA, operates 450 hours at max; iPhone 8 can last about 13 hours with 1,821mAh battery)
Power by wall charger	no limits in power consumption, can use WiFi, motor, etc	fixed position
WiFi	wider range (150m), faster transfer rate (1Bbps)	consume more energy (350µW to transmit data)
BLE	longer battery (25.5µW to transmit data), greater mobility	shorter range (10m), lower data rate (2Mbps)

3.3.3 Hub

The hub is a device which can communicate other connected toys, the collar or the mobile application. Also, it could be able to connect to the cloud to exchange data.

To that extent, the hub should be able to access to the Internet via a dedicated WiFi or wired interface. For contacting with the toys or mobile application, it must use the same wireless technology as that of those devices (either BLE or WiFi).

The hub should have its own software system to manage and exchange with other devices, such as triggers a nearby toy (e.g., turn on/off) or receiving commands from the mobile application. It should also have a data processing application to extract insights from the downloaded sensor data from the collar or others (e.g., the pet is active or idle too much). Commonly, the hub is designed to be fixed at a location and powered by a constant electric source. Therefore, it does not have tied constraints on physical shape, power consumption and hardware parts. It has a wider range of selections for operating systems, memory and storage size, could have both BLE and WiFi at the same time (Table 3.3).

In prototyping procedures, we suggest using a computer as a hub. For example, Intel Mini PC[1] and Raspberry Pi 4 Module B[2] are suitable candidates. They can run a mainstream operating system (e.g., Debian) and have hardware modules for WiFi and BLE communication.

3.3.4 Cloud applications

In this system, cloud applications are the software components run at the external server, usually utilizing the cloud computing technologies which is usually a remote computer in the Internet or an in-house server. The comparison of these two approaches is outlined in Table 3.4.

These applications can work as a data storage and processing software for all collected data from other devices. This cloud application could perform large data analysis to detect abnormal

[1] https://www.intel.ca/content/www/ca/en/products/boards-kits/nuc.html

[2] https://www.raspberrypi.org/products/raspberry-pi-4-model-b/

Table 3.3 Hub requirements

	Requirements	**Notes**
Communication technology	major standard communication technologies (e.g., WiFi and/or BLE)	WiFI/BLE to connect with connected devices; WiFi to connect with the cloud applications
Energy source	constant energy sources	using wall charger
Operating system	should run full operating system (e.g., Debian, Windows)	unitize existed software stacks (e.g., Mozilla Things, ThingsBoard)
Hardware	no fixed constrains	reasonable for domestic spaces
Physical shape	no fixed constrains	reasonable for domestic spaces

Table 3.4 Simple comparison between cloud provided and in-house server

	Pros	**Cons**
Cloud-provided server	• not need to prepare physical devices and infrastructure (e.g., renting from Azure, Amazon) • can add more storage and server easily (e.g., by renting) • can access data anywhere via Internet (e.g., via web-browser, remote terminal)	• depends on Internet access (e.g., max data transfer speed) • depends on provider's technologies and tools (e.g., cloud management tools of Amazon)
In-house server	• has physical control the hardware for trouble shooting and backup • keeps critical data in house • not rely on Internet to access the data	• needs prepare physical infrastructure for the server to backup (e.g., router, Internet line) • can lose data if severs crash

behaviors or produce recommendations based on well-being data of the pets. It must be able to synchronize data with the mobile phone, the hub or other Internet-connected devices in the toy system.

In addition, the cloud software plays the roles of a command proxy to relay information from the mobile application to the hub, the connected toys and collar in the system. One example of this usage is the scenario 3 Section 3.2.5.

In summary, the cloud software part should include databases for sensor and other data, a message relay service and a data analysis part (Figure 3.8).

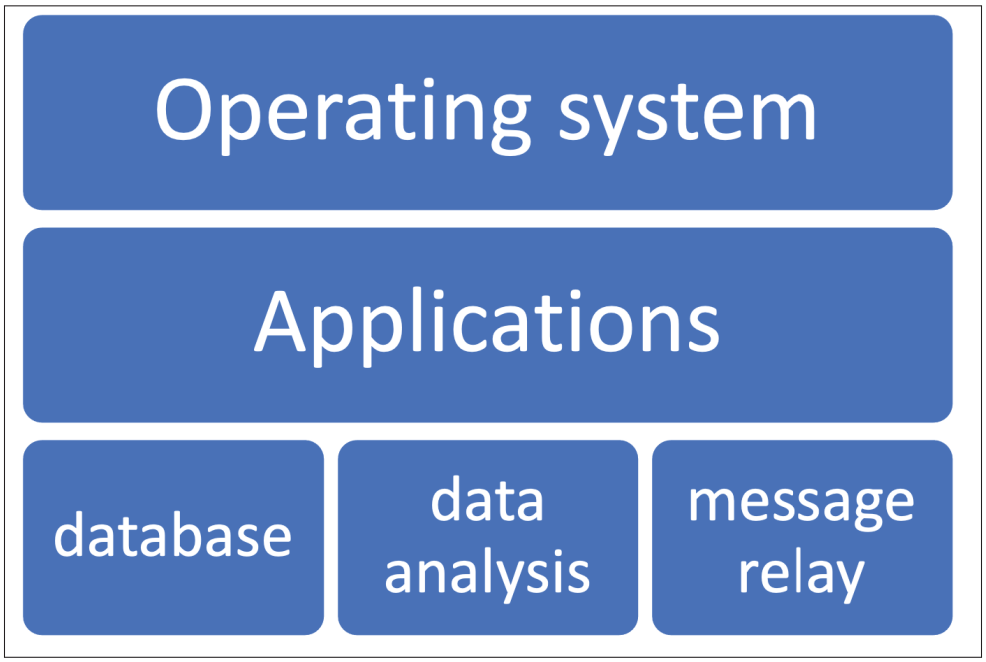

Figure 3.8 Cloud applications

3.3.5 Mobile application

The mobile application is the main interface that interacts with the user. It could acquire data directly from the collar and/or toys to analyze, receive data from the cloud for displaying or synchronize information with the cloud part. For example, to be able to display a pet behaviour timeline, the mobile application acquires and displays the analytic results from the analytic software of the cloud to the owner. The mobile application also does data analysis on its own with a small number of sensor data gathered from the collar or connected toys. The mobile

application could send commands to the cloud, which then forward to the hub, the toys or the collar to execute certain tasks such as triggering the food dispenser.

The mobile application should run on major mobile operating systems and provide the following features:

Table 3.5 Main requirements for the mobile application

Requirements	**Notes**
runs on major mobile operating systems	Android and iOS
collects the data from the collar and/or connected toys when in range	depends on system architecture
analyzes the pet behaviors based on collected data	uses small amount of data (e.g., if the phone has 500MB of free space, max data can be used is 500MB)
communicates with the hub for exchange data and commands	depends on the system design

3.3.6 Data storage strategy

Raw data of the system comes from the collar sensors, from other connected toys or devices. When that amount is relatively small compared to devices' memory, they can be stored inside the collar or the toy flash drive, inside the mobile phone application or the hub.

However, over time, the accumulated data can be out of the storage capacity of the devices.

For example, assume that we have a collar with one magnetometer, one accelerometer and one GPS sensor. The magnetometer and accelerometer have sampling rates at 50Hz and that of the GPS is 1Hz. We assume that each data record contains 4 bytes for timestamp, 12 bytes for accelerometer data, 12 bytes for magnetometer data and 92 bytes for GPS data (Table 3.6).

Table 3.6 Sensors and their data sample size

	Sampling rate (Hz)	**Data size (byte)**	**Timestamp size (byte)**
Magnetometer	50	12	4
Accelerometer	50	12	4
GPS	1	92	4

In general, a collar with n sensors generates an mount of $data$ in a period of $time$ can be calculated by this equation:

$$data = \sum_{i=1}^{n} rate * (size + timestamp) * time \tag{3.1}$$

with $size$ is size of data in each sample of the sensor i, and $timestamp$ is the number of byte to store the timestamp.

By using the equation, we can measure that for each second, there is 1696 byes stored. For 24 hours, that is about 140. After one month, the amount of data is roughly 4GB. After 356 days, that amount is nearly 50GB.

If the collar has 8MB of flash memory, it can store data of nearly 1.5 hours. The mobile phone with 32GB memory can store data of less than 10 months.

Furthermore, there are risks of losing data if the data in the mobile phone or the collar is corrupted.

Because of these reasons, the data should be periodically moved from device memory to the cloud database such as using a relational database management system (RDMS). Take the example of PostgreSQL, it has no limit of database size, while MySQL limits it to 256TB.

Table 3.7 presents the data generated by a collar with 8MG flash memory within several time frame (1 hour, 2 hours, 1 day, 1 month, 7 months and 1 year). This collar includes one accelerometer, one magnetometer one GPS sensor with sampling rates are 50Hz, 50Hz and 1Hz, respectively. The table shows that the collar can only store less than 2 hours of generated data.

It is fast to access data inside the collar but has risks of data loss. A phone or hub with 32GB memory can store more than 7 months of data while a database can store up to unlimited data. In these two cases, it is optimal to store a larger amount of data. Also, the data access speeds are slower and depend on Internet infrastructure or database structure.

Table 3.7 Data generated by a collar (one accelerometer, one magnetometer one GPS sensor and 8MB memory; the magnetometer and accelerometer have sampling rates at 50Hz; the GPS sensor sampling rates is 1Hz)

Data generated (size/time)	**8MB memory (collar)**	**32GB (phone or hub)**	**Database**
5.8MB/1 hour	yes	yes	yes
11.06MB/2 hours	no	yes	yes
139MB/1 day	no	yes	yes
4GB/30 days (1 month)	no	yes	yes
28GB/7 months	no	yes	yes
49GB/356 day (1 year)	no	no	yes
Pros	instance data access and easy to store	can store data around 7 months	large data storage (up to 256TB or unlimited)
Cons	loss data if memory corrupted	slower data access via BLE/WiFi (2Mbps – 1Gbs)	depends on Internet and database infrastructure

3.3.7 Data analysis

This component utilizes the collected data from collars and other toys to prediction and summary the pet well-being, for example based on the accelerometer data, it could predict the pet behaviors and provide the owner recommendation actions to improve or maintain the physical state of the pet.

This analysis might happen in these places of the system: on the collar, in the mobile application/the hub, and in the cloud applications.

1. Behavior analysis on a mobile application or the hub: it uses the aggregated data on a short period for the analysis (e.g., around 1 to 2 hours before the collar's 8MB memory is full).

For example, the hub collects all the data from the collar when the pet passes by it, which is happened between short periods (e.g., every 2 hours).

In addition, if the labeled data are given to build the analysis model, the supervised machine learning method is used. It can classify behaviours such as sleeping, resting, walking or running.

2. Behavior analysis on the collar: the previous approach has some limitations such as requiring relatively more storage memory for raw sensor data. To produce faster analysis results, we propose an analysis application running inside the collar. It could use the real-time sensor data to instantly issue the prediction about the pet actions right at that moment.

   In this way, it is possible to store prediction results inside the collar and send them out to external applications (e.g., mobile application). This would significantly reduce the data storage needs for the collar (e.g., 1 byte for a prediction result versus 20 bytes for each accelerometer record).

   However, this approach poses two problems: 1) this kind of embedded software could consume an extra computing resources (in settings of (García-Martín, Rodrigues, Riley & Grahn, 2019), it costs 512J to run Random Forest algorithm) and 2) requires suitable technologies to update the embedded applications (e.g., upgrade firmware over-the-air - OTA or via a cable).

3. Behavior analysis at a potential computing device (e.g., cloud): The two previous approaches have the limitations in CPU (e.g., optimal for low-power, energy-saving purposes) and storage capacity (e.g., 16GB - 256G). Assume that the collar collects sensor data for a long time (like 1-2 years - which might be around 44GB - 88GB), it is not efficient to put those data inside a smartphone or a hub. Also, perform analysis at such relative a huge quantity data points are not optimal on low-power devices.

   Because of that, another external module that performs data analysis is needed. It could be installed in a remote computer in the cloud. Then, the results from this module could be sent back to the phone or hub for further procedures.

   In prototyping processes, this computing device could a cloud computer machine (such as an Amazon EC2 or an on-premise server).

Table 3.8 summarizes the pros and cons of each approach.

Table 3.8 Comparison of different data analysis approaches

Place	Pros	Cons
Mobile application or hub	• analyze recent collected data (e.g., within 1-2 hours) • provide immediately results after analysing	• limit by computing resources (e.g., 1GB memory card) • can analyze only available data on the device
Collar	• provide in device real-time analysis • reduce data storage size (e.g., 1 byte for each predicted point vs 20 bytes for each accelerometer record);	• might need more computing power and resource optimization algorithm ((García-Martín *et al.*, 2019) 512J to run RandomForest algorithm, a 1000mAh battery with 3.6V has energy to run it 10 times)
External computing device	• analyze large data quantity in terms of time and size (e.g., 1 year data - 44GB)	• require Internet access • response time depends on Internet acess conditions

3.4 Proposed system architecture

Based on previous technical analysis in Section 3.3, it is found out that there are two major challenges to build a smart toy system for pets. The first one data storage and the second is power consumption.

In this section, we propose the following possible system architectures to build such system. At the same time, we investigate which solution is more viable than the others when tackles the above two challenges.

3.4.1 Architecture 1

In this system architecture, the collar plays as a central for all toys and connected devices. In addition, it is possible to be integrated into existing smart home system such as Google Home (means that Google Home applications can interact with the collar directly). The overview of this system design can be seen in Figure 3.9.

The roles of each component and its functionalities are detailed as below.

- The toy is a WiFi connected device, which communicates directly with the collar only. It sends the data to the collar and can be triggered by the collar.
- The hub acts as a middleman application and temporary data storage when the owner does not have the mobile application nearby. It connects to the collar to acquire the collected data and synchronizes this information with the cloud server. The pet owner can send commands to the hub via the Internet. Those commands are then forwarded to the collar and can activate certain toys if needed.
- The mobile application: is the main user control application. The pet owner uses it to see information about their pet well-being analysis status. The main functions of the application are to trigger commands in the collar and send/retrieve data from the cloud server.
- The cloud: is the main data storage and data processing unit. After having historical data from the pet behaviors, the analyzing procedures are executed in the cloud to perform activity prediction. The results are transferred to the mobile application for presentation to end users. The cloud also is a proxy to exchange information between the hub and the mobile application, such as sending commands.
- The collar: is attached into the pet body. It contacts other toys, the mobile application and the hub via WiFi. It can activate the toys indirectly via command from the mobile application or the hub.

This architecture is optimal for fast data transfer rate and wide coverage area by using WiFi technology. However, it requires appropriate hardware designs for each device because power consumption is much higher than using BLE. This architecture also has a single point of failure:

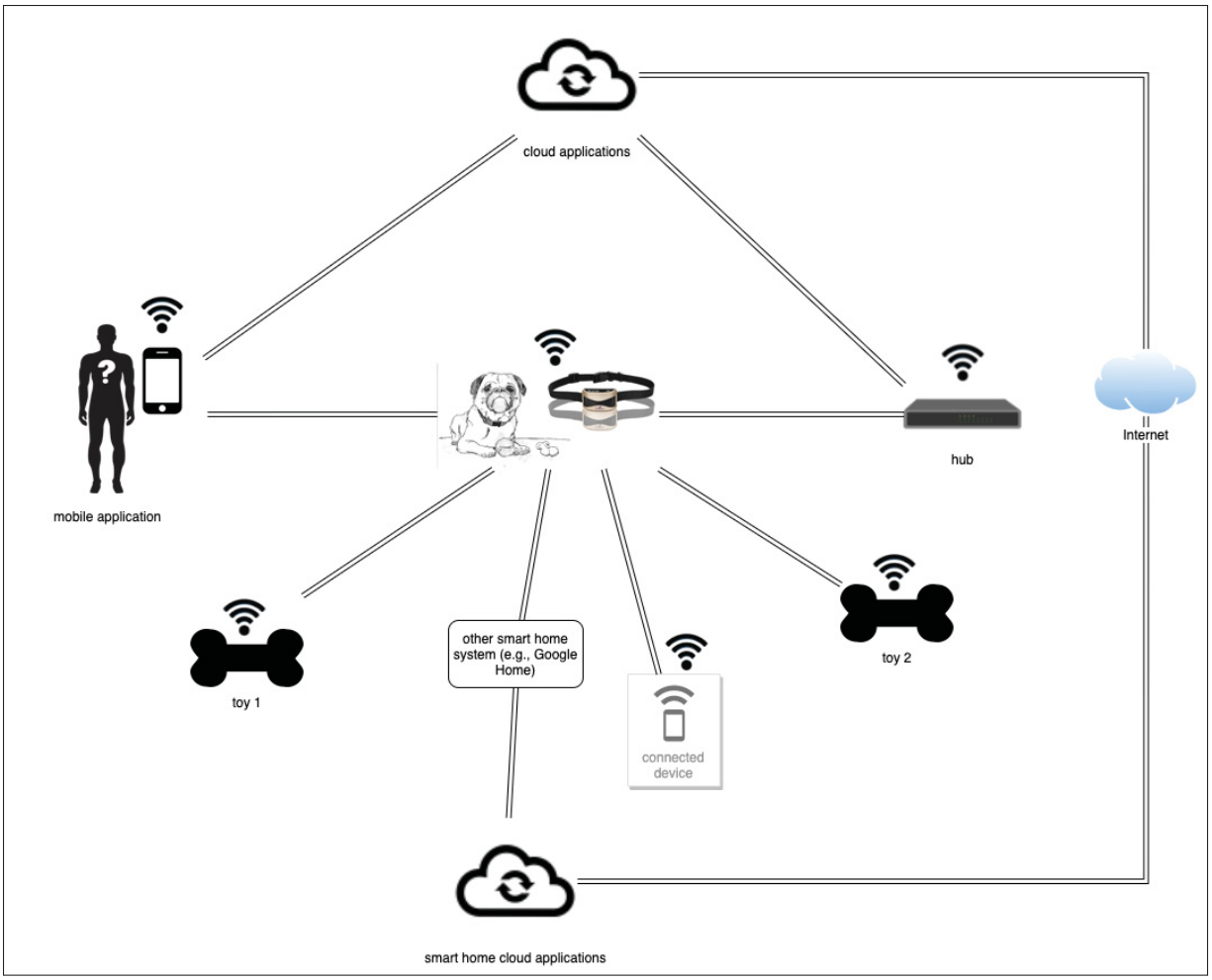

Figure 3.9 System design based on architecture #1

the collar. If the collar is damaged, the whole system cannot function anymore. The mobile application and the hub are still able to communicate via Internet but they cannot talk with any other toys or connected devices. We summarize the pros and cons of this architecture in Table 3.9.

3.4.2 Architecture 2

This system architecture is optimal for devices having low power capacity. The mobile application and the hub are the only points that could access to the Internet or talk to each other via WiFi networks. All the internal communications inside the system are operated under BLE technology.

Table 3.9 Summaries of the architecture #1

	Pros	Cons
Toys or connected devices using WiFi	• bigger connectivity area than BLE (150m vs 10m) • faster data transfer (1Gbps)	• consume more energy to run WiFi • can affect the operation time of battery power toys/devices • depend completely on the collar
Hub	• less complex in terms of application requirements (only communicate with the cloud and the collar)	• None
Collar	• wider connectivity range • faster data transfer speed • can contact multiple toys/devices at the same time • easier to integrate into existing smart home system	• is the single point of failure: if the collar is disconnect, the system cannot function • require much more battery capacity and physical size

The mobile application and the hub can contact with the collar, the toys and other connected devices to get the data or control them.

The overview of this architecture can be seen in Figure 3.10.

The roles of each component and its functionalities are detailed as below.

- The toy is a BLE connected device, which communicates directly with the collar or . It sends the data to the collar and can be triggered by the collar.
- The collar, the toys and connected devices: have the equal roles in the system. They can communicate only with the hub, or mobile application vie BLE. They all send the data to the mobile application/hub and can be controlled by them only.
- The hub has the same functions as in the architecture #1. However, it can control and get data from all the devices such as toys or the collar via BLE.
- The mobile application is similar to that of the architecture #1 and uses BLE instead of WiFi.

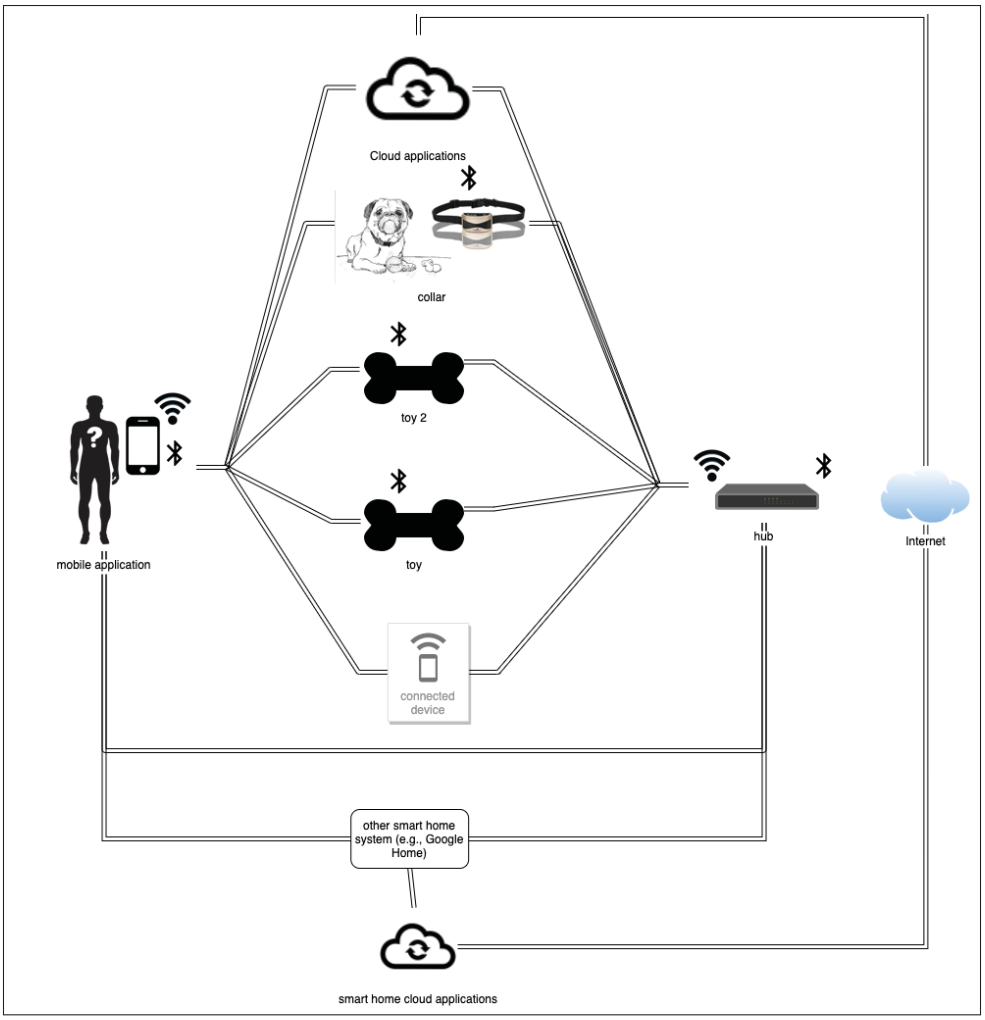

Figure 3.10 System design based on architecture #2

- The cloud plays the same roles as in the architecture #1.

In this architecture, the mobile application and the hub is the central points of the system and have very similar functionalities. They both can control the collar and other connected things (e.g, toys) and can talk directly to each other for data exchange. In addition, they both can implement the integration abilities to existed smart home systems (e.g., Google Home). If one of them is disconnected, the system can still operate normally by using the other one. In this design, the collar and other toys have equal roles. They all connect to the hub or the phone

via BLE which limits limit their connectivity range. However, in this way, these devices can conserve energy better than that of architecture #1.

The technical choices for this architecture can be seen in Table 3.10.

Table 3.10 Summaries of the architecture #2

	Pros	**Cons**
Toys or connected devices using BLE	• require less energy, operate longer time • can be lighter due to reduce battery size • can contact to both mobile phone and the hub	• shorter connectivity range • can effect the function time of battery-powered toys/devices
Hub	• can control all BLE device and contact with the mobile application • can integrate with smart home systems	• require a more complex software to manage all connected devices
Collar	• longer battery life • simpler application functions	• slow data transfer rate • shorter connectivity range

3.4.3 Architecture 3

This system architecture is optimal for inter-operability and extensibility. Any compatible protocol WiFi-connected toys can be added into the system. At the same time, it moves the failure point to the Internet infrastructure. In this design, all devices connect directly to the Internet via WiFi. All data exchange operations are going through the cloud applications (Figure 3.11). Similar to architecture #1, the collar has applications to integrate into existed smart home systems.

In additional to having the same disadvantages when using WiFi connection for the collar, toys and connected devices, the function of this system depends on the cloud and common Internet

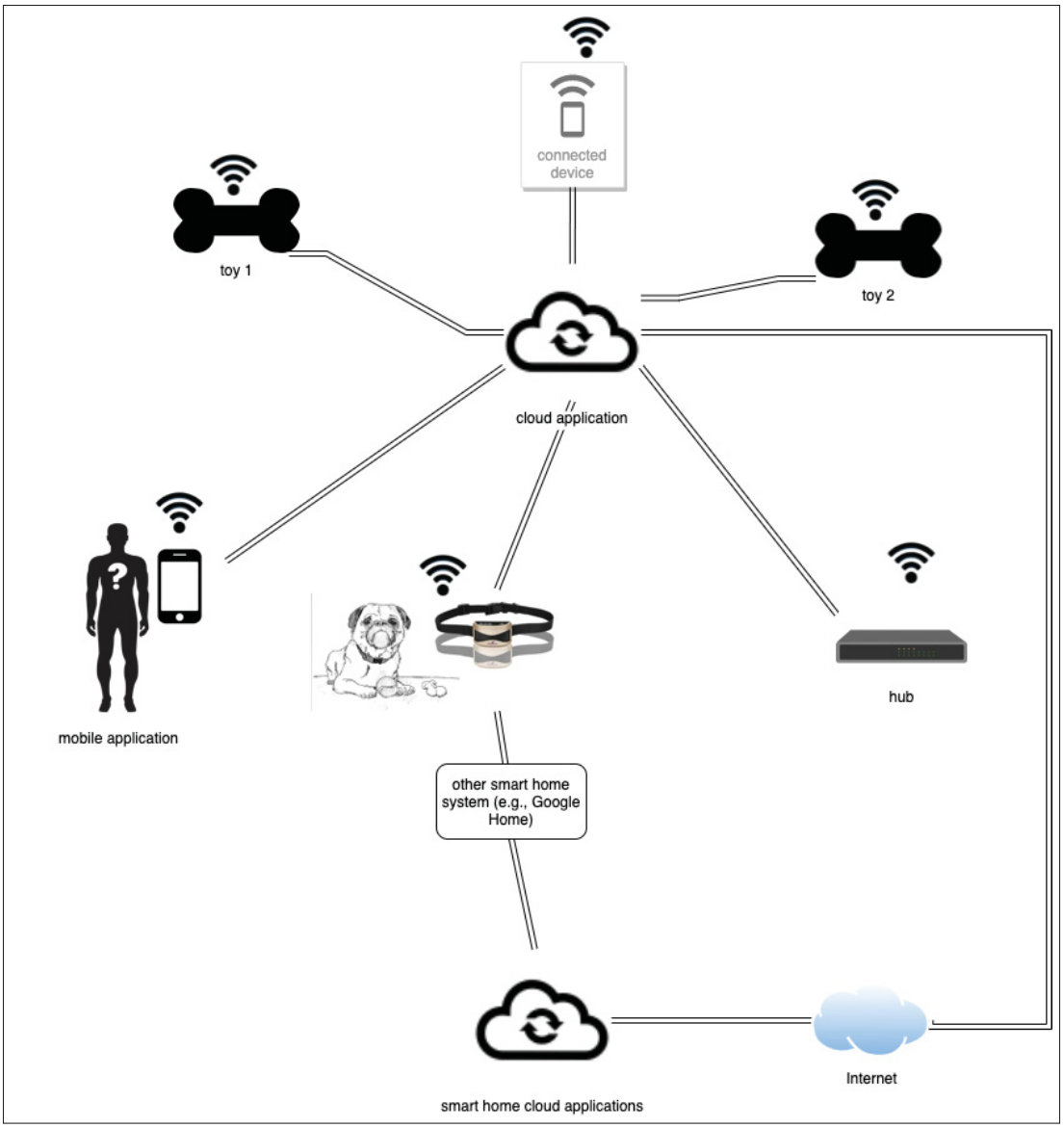

Figure 3.11 System design based on architecture #3

infrastructure provides (such as Amazon, Azure). This is both the biggest advantage and the biggest disadvantage.

Due to using Internet access, each part of the system can operate independently regardless of other disconnect ones. For example, the collar can still fully function when all other parts of the system, except for the cloud, are down. It is the same for the toys and connected devices. In detail, we make a table for technical options for this architecture.

Table 3.11 Summaries of the architecture #3

	Pros	**Cons**
Toys or connected devices using WiFi	• faster data transfer rate • longer connectivity range • can connect with all other devices in the system	• consumes more energy than using BLE • has shorter operation times when using battery
Hub	• can communicate with all devices and appliactions in the system	• requires more complex software to manage all connections
Collar	• has wider connectivity range • has faster data transfer rate • can better integrate into the existed smart home systems	• consumes more energy • might need bigger dimension for battery

3.4.4 Comparison with the Google smart home architecture

In order to validate our system architecture, we compare their designs with a smart home architecture from Google patent (Chen, Li, Hsieh, Xiao & Shang-Hui, 2014) (see overview in Figure 3.12).

In that patent, home appliances (number 30) are similar to collar, connected devices and toys in our systems while the smart gateway (number 20) has some functions of the hub. The hub remote controller (number 10) has some functions of the mobile application because it can send control command to the hub. However, the internet controller (number 34) is on par with the mobile phone application and the server (number 33) is the same as the cloud part.

This smart home architecture has more components than our proposed system. Also, it treats all smart home appliances in the same ways, which is similar to the architecture 2 (collar or toys have equal roles). However, all the home devices are connected to the Internet as in the case in the proposed architecture 3.

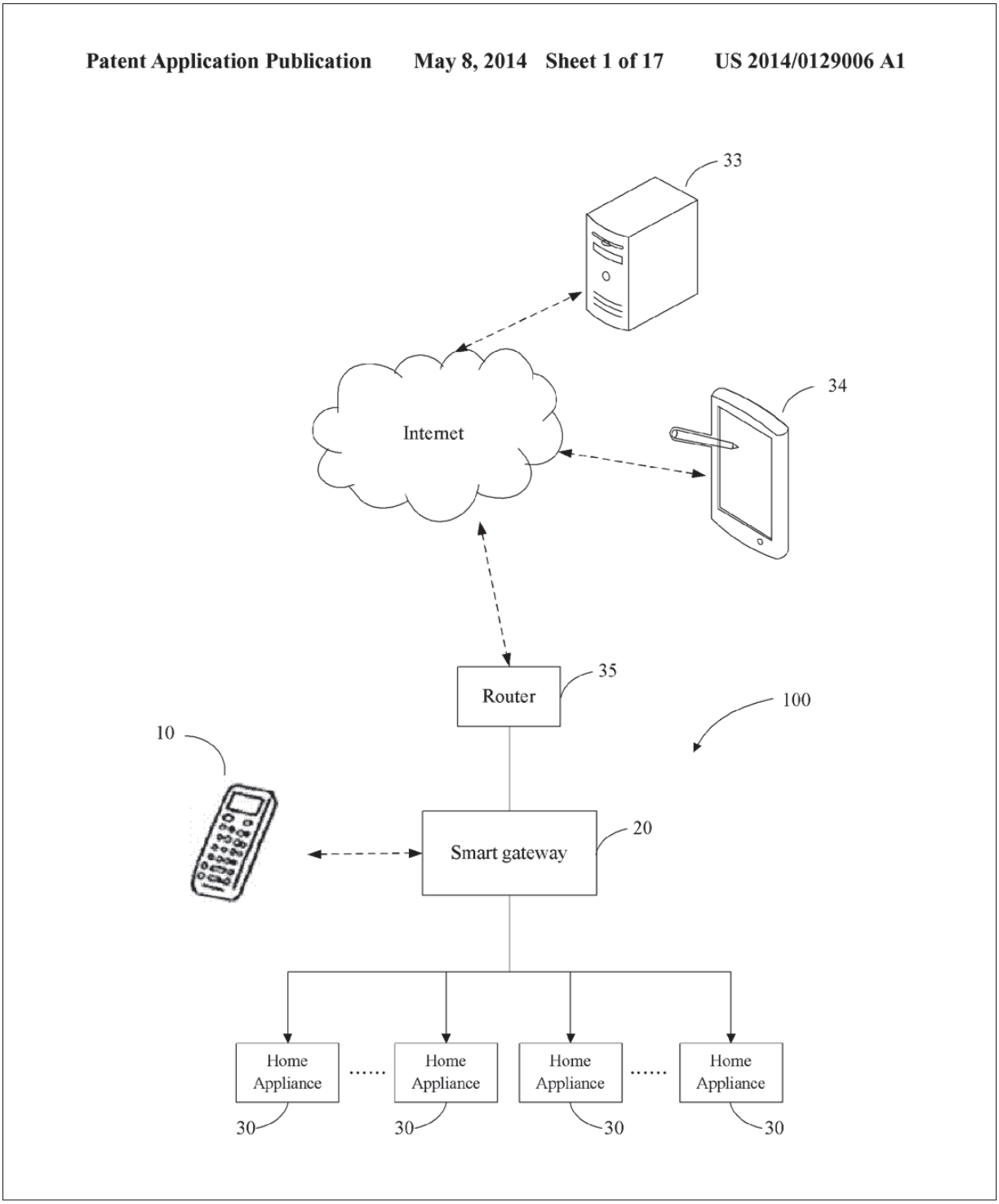

Figure 3.12 Smart home architecture of Google
Taken from Chen et al. (2014, p. 2)

In general, this architecture and our proposed ones have many similarities at the abstraction level. Table 3.12 presents a comparison on advantages and points of failure of this smart home architecture system and our proposed architectures.

Table 3.12 Comparison between architectures

	Architecture			
	#1	**#2**	**#3**	**Google**
Advantages	fastest data transmission data	optimal for low power toys, devices	availability and fast communication	fast connection and availability
Points of failure	collar	mobile application, hub	cloud application, internet	Internet and cloud applications

All the 3 architectures can tackle the problem of data storage. Architecture 1 has a single point of failure, which is the collar. Architecture 2, 3, and Google are better in terms of risk mitigation by distributing failure points at two locations. However, only architecture 2 can minimize the problem of energy consumption. Hence, it is the most viable solution for this project.

3.4.5 Prototype architecture for demonstration purposes

There are several factors that are hard to estimate such as power battery life, factual performance (e.g., speed, time) of real device when running. This leads us to design another simplified system to implement for demonstrating purposes (Figure 3.13).

In this prototype system, it is based on the design of system architecture 2 to tackle the most two challenge problems: energy consumption and data storage.

Because of that, we only implement the crucial components such as the mobile application, the collar and the cloud application.

In this system, the mobile application controls and collects data from the collar via BLE. It then performs data analysis on that data segment to detect current pet behaviors. The data are sent to

the cloud - via WiFi - for storage and analysis to produce an overall behavior picture of the pet over time.

In addition, to examine the system performance when running sophisticated analysis algorithms, we use machine learning approaches to predict the pet activities. The details of this system and how it is developed is describe in the next chapter.

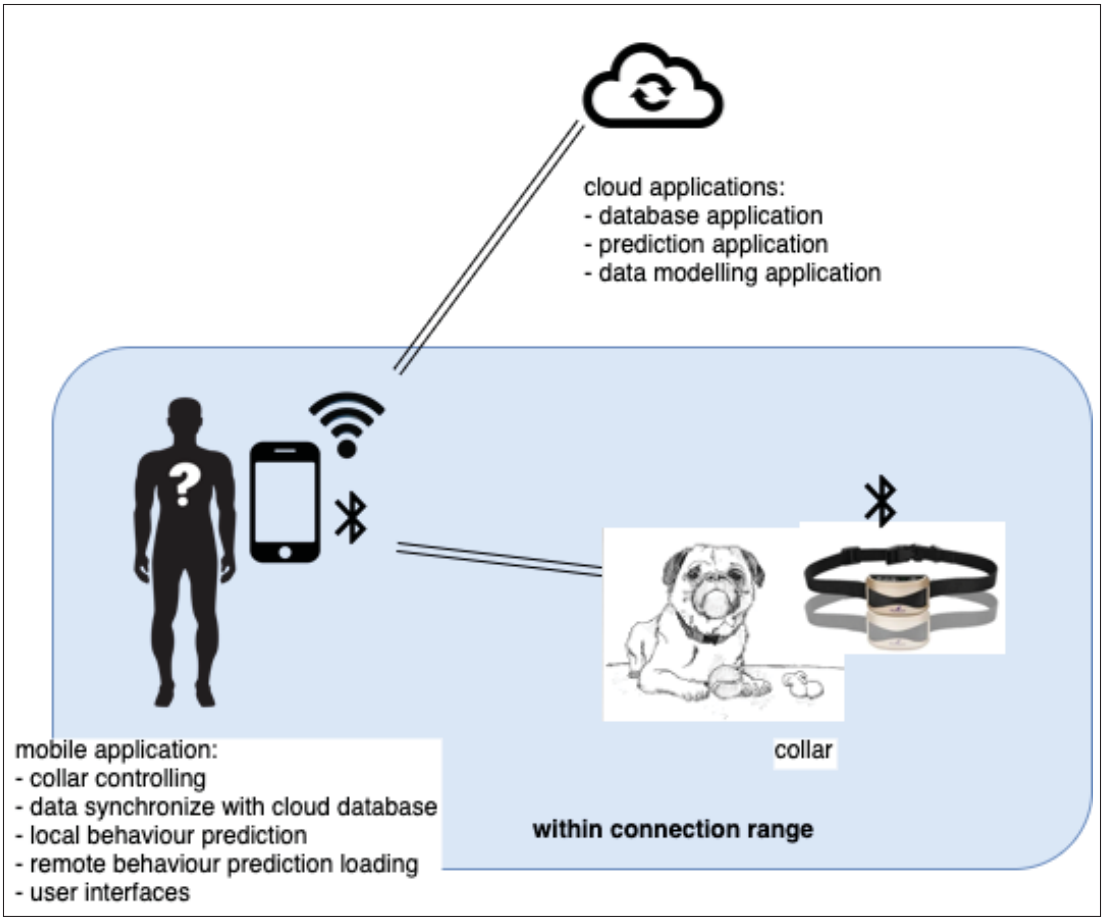

Figure 3.13 Architecture of the demonstrated system

Before implementing the system, we define certain criteria that help to guide the development. These criteria are also used to validate the results of the system at the end.

- The mobile application is functioning as designed and runs on real iOS devices.
- It is possible to store a relatively large amount of accumulated sensor data over time.

- The behavior recognition model is integrated into the mobile application.
- The behavior recognition component can be applied to accumulated sensor data and run on an external server.

3.5 Chapter summary

In this chapter, we proposed the working methodologies to complete the smart-collar-communication system. We suggested system architectures with advanced analysis applications to deal with sensor data. As well, the architectures were able to adapt to multiple usage scenarios: from indoor to outdoor. Based on our design, the system is robust and possible to add multiple devices. Not only toys but also other connected devices (such as Bluetooth beacons) could be a component of this system. We also proposed a simplified system for demonstration purposes. In the next chapter, we will go into more details of the implementation of the proposed system.

CHAPTER 4

DEMONSTRATION SYSTEM DEVELOPMENT AND DISCUSSIONS

4.1 Introduction

In this chapter, we implement the ideas of building a system of smart toys for pets with behavior recognition abilities. This prototype system is based on the proposed architecture in Section 3.4.5. In general, the implemented system is a proof that it is possible to solve two main challenges that are data storage and energy consumption.

In detail, we present an overview of what parts of the system are implemented. Then, we go into details about how each part is created. Also, we discuss the implementation of mobile application and the machine learning analysis part as well as how they are connecting via the external clouding computing components.

It is worth noting that the collected data of this system is limited in two pet behaviours: stationary and moving. It is limited because the data collection steps are conducted on real animals and provided by BeOneBreed. This research does not include data acquisition from real pets or testing on them.

1. **For data storage**: we use the in-application memory as a temporary data storage before moving all data to a database system. By doing this way, the collar has enough space to storage because it can delete all recorded data after sending them to the mobile phone. At the same time, the mobile application does not depend on the storage capacity of the phone by sending out all of its data to the PostgreSQL in the cloud.
2. **For energy consumption**: In order to reduce the energy consumption, we propose two potential solutions. First, we use BLE as the communication method between the mobile application and the collar as it is proven to be an energy-saving technology (see more at Section 2.2.2). Secondly, we use in-cloud data analysis approach (e.g., AWS) to off-load the computing-intensive tasks from low-power devices (the mobile phone and the collar) to the remote computer.

Here are the results of the implemented system:

- **The system**: is implemented as in its architecture. The collar collects sensors' data. The mobile application retrieves those, performs classification, and uploads them to PostgreSQL; the cloud data analysis applications pull all sensor data from the database, and executes behavior prediction. After that, the predicted results are put on the internet. The mobile application then retrieves and displays them to the user.
- **The collar**: is a functional prototype. We use this collar as the base to propose other connected devices and software. This early prototype is fully working but has certain limitations, which is discussed later in Section 4.5.
- **The mobile application**: is fully developed. It has the abilities to control the collar, such as turning on, and off the sensors. It also provides the basic functions to erase the collar's flash memory and download the sensor data to the phone. This application also uses the machine learning model to predict the pet behavior. Moreover, it has a function to upload sensor data to a remote database via the internet. In addition, it can fetch the predicted behavior results from an Internet source and show to users as a visualization.
- **Data**: are temporarily stored inside the collar before transferring to the mobile application's internal memory. In the phone, data are stored there for a short time before being uploaded into a remote PostgreSQL database via the internet. It is also worth noting that there is a mechanism to transform raw sensor data into a common format is introduced.
- **The data analyses**: are run in the cloud (for all data) and partly in the mobile application (for temporary collected data). To select a suitable algorithm, we run multiple machine leaning modelings on the data and select the best model. It is shown that Random Forest classifier yields optimal outcomes. It is used for inferring tasks in the mobile application and the cloud.

We also published the preliminary results of this study in Proceedings of the Canadian Society for Mechanical Engineering International Congress 2020 (Hof & Hoang, 2020) (see Appendix I).

4.2 Early prototype of the collar from BeOneBreed

This preliminary collar prototype (Figure 4.1) contains the nRF52832 ARM Cortex-M4F processing unit in the form of the BC832 miniature module from Fanstel. The outer case components are produced by a 3-D printer. The collar features a wireless charging battery pack. Its weight is about 50 grams and has one accelerometer sensor. The collar has 8MB internal memory to store data (developed by Motsai[1]). It is equipped with a BLE 5.0 chip. Table Table 4.1 provides the collar specification in detail.

Figure 4.1 The preliminary prototype collar

[1] http://motsai.ca

Table 4.1 Collar specifications

Feature	Specification
System-on-Chip	nRF52832
CPU	ARM Cortex-M4F
RAM	64kb
Memory size	8MG
Sensor	Accelerometer
Battery life	1 hour continuously at full power
Weight	50g
Communication	BLE 5.0

4.3 Mobile application

The mobile application is developed for iPhone and compatible with the iPad (for iOS version greater than 12.0). iOS is selected due to its popularity in the world (2nd most popular smart operating system, just after Google's Android operating system[2]). Also, by choosing iOS, we can reuse parts of the third-party Bluetooth communication libraries that are developed by others.

The mobile application is developed using Swift programming language, and optimized for devices having screen size of 4.7 inches. The software also uses some Apple's technology such as CoreML to execute the machine learning model. The source code is publicly accessible at GitHub (see II).

An overview of the mobile application structure can be seen in Figure 4.2. Each function will be discussed in detail in the following sections.

[2] https://www.idc.com/promo/smartphone-market-share/os. Accessed Sept 1 2020

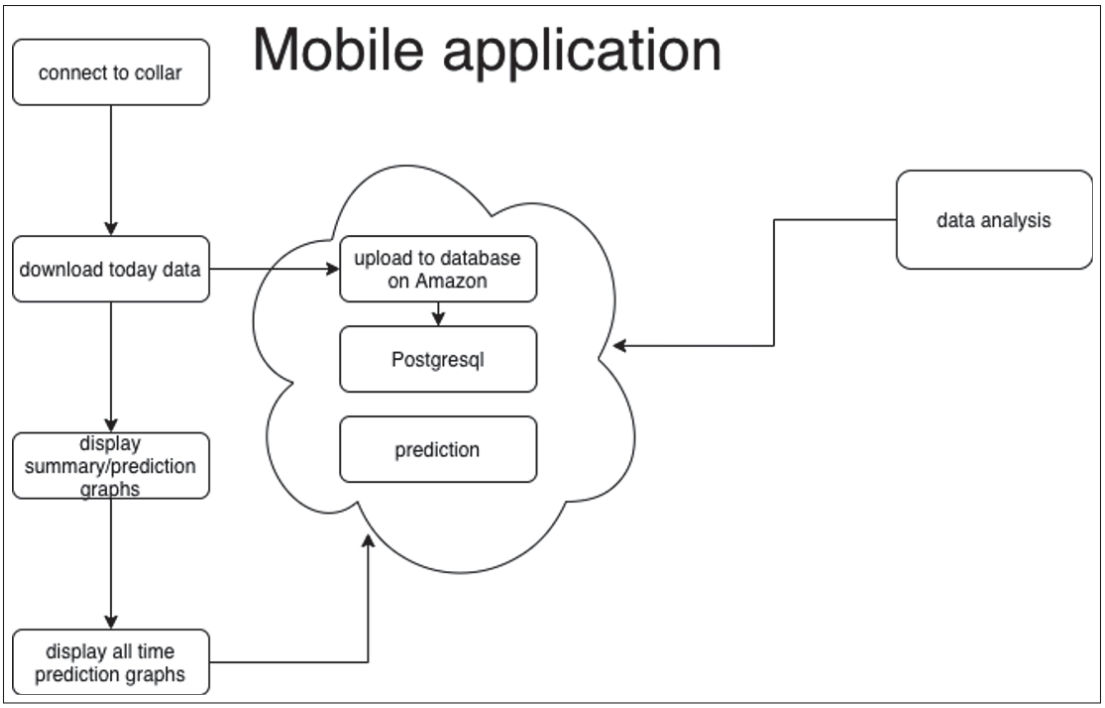

Figure 4.2 Overview of mobile application functions

4.3.1 Technical issues in development

When building the mobile application, we encounter several practical challenges.

Firstly, BLE functionalities on iOS devices are not consistent. BLE functionalities do not work on the emulator so all the tests are operated on the real collar. However, the collar functions are not stable, it must be turned on and off several times before we can confirm that it is running or not.

The BLE communication library provides by Apple also behaves differently depends on the iOS devices. For example, with the same configuration, the mobile software for mobile phone A can connect to the collar via BLE while that of mobile phone B cannot.

The differences in iOS version also create weird behavior. We accidentally update one mobile phone to the latest iOS version and the application crashes. It operates again after Apple releases

a new update. We had spent months on these issues without specific solutions besides the trial-and-error method.

Secondly, the sensor data format is beyond our control. This format is created at the firmware level, which is created by a hardware vendor. Because of this, we cannot optimize the size of data stored on the device's memory. The suitable approach for the project is to keep the current structure and convert it to another type when needed.

Thirdly, it is noted that BLE commands are not executed at the time as they are sent to the collar. It takes a certain amount of time to execute them. For example, on the software side, the 'erase' command is triggered then the 'record' is triggered later. Both are successfully executed without error. However, in the real device, the 'erase' command is still working and not yet finish.

4.3.2 User interface

We design the application for mobile phones with the screen size 4.7 inches (such as iPhone6/7/8/SE 2nd). The application can be developed in Xcode 11, programmed in Swift language and its user interface is based on a storyboard [3]. Xcode supports a second way to build the user interface called SwiftUI [4]. It is promoted for more than 1 year and is going through major changes every version which requires rewritten source code. Therefore, we choose to use storyboard because of this maturity and stability. The storyboard design of the mobile application can be seen in Figure 4.3

[3] https://developer.apple.com/library/archive/documentation/General/Conceptual/Devpedia-CocoaApp/Storyboard.html

[4] https://developer.apple.com/xcode/swiftui/

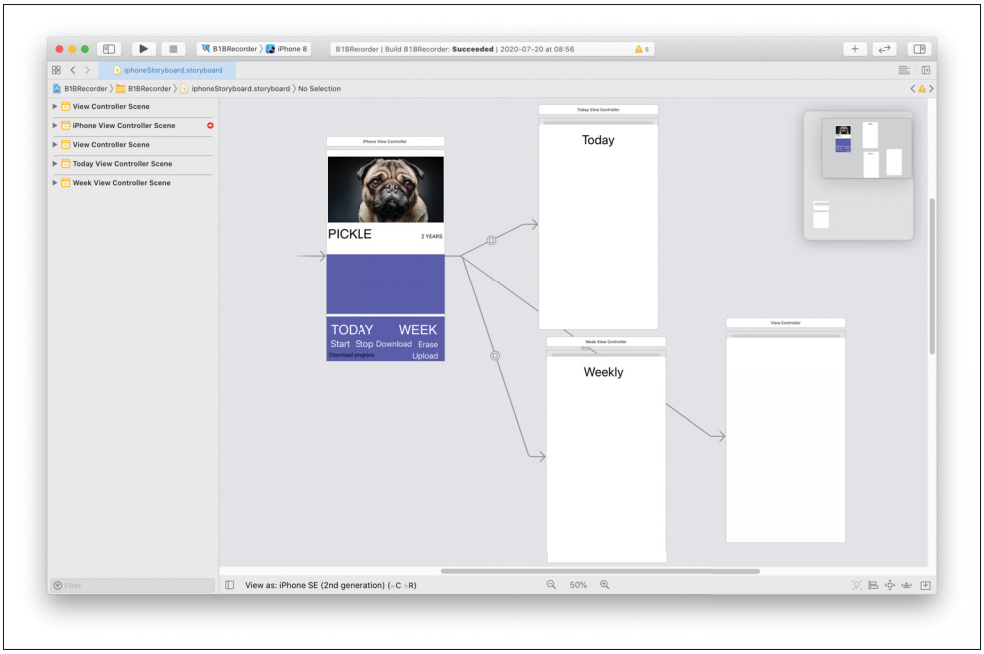

Figure 4.3 Storyboard of the mobile application shown in Xcode

Because of the small screen size on mobile phones, we split the functionalities of the application into three screens. There are one starting screen and another two ones for display graphs.The whole application flow can be seen in Figure 4.4.

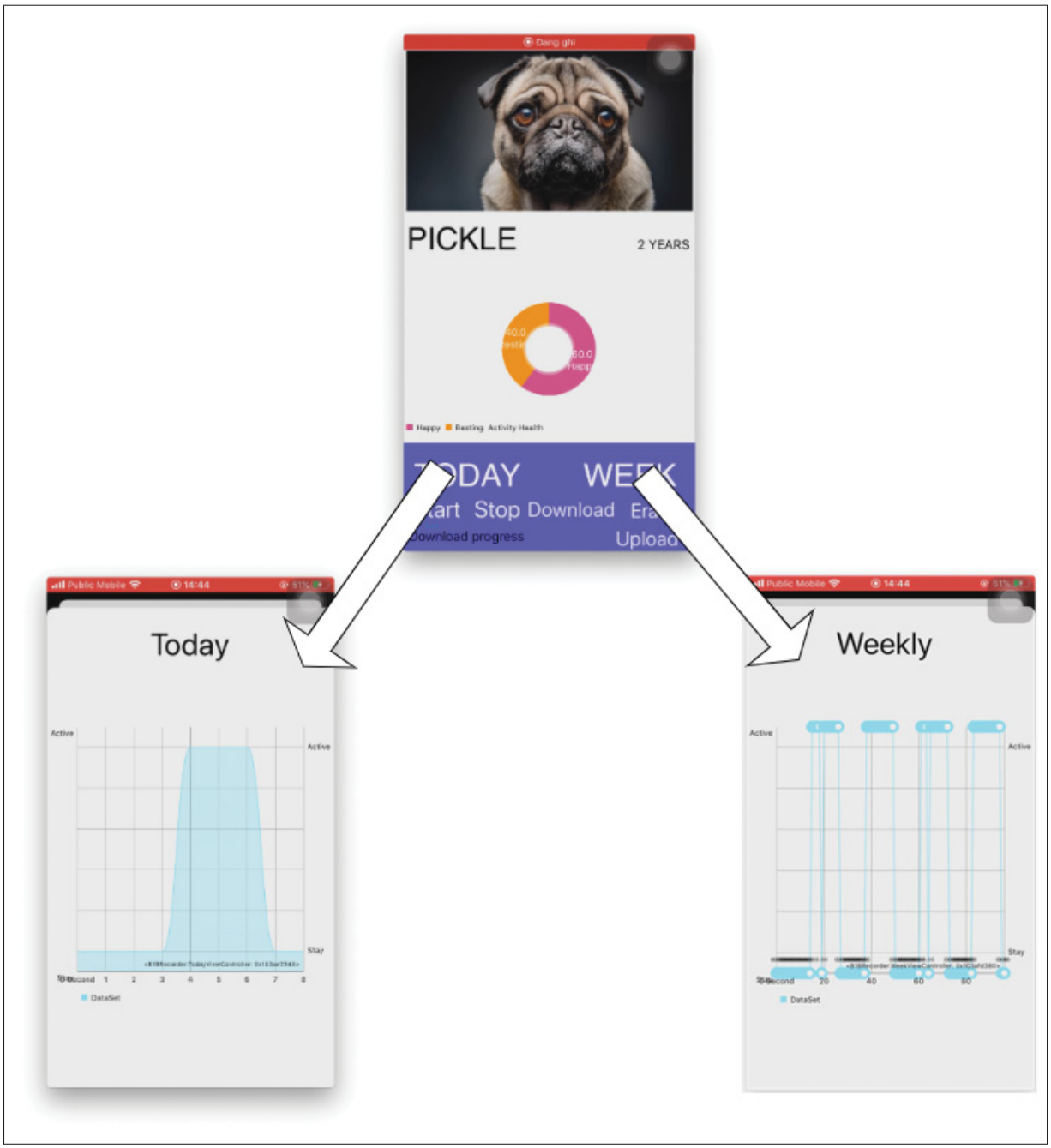

Figure 4.4 The application with three screens: one launching screen and two for graph displaying

Figure 4.5 is the starting user interface of the mobile app. It displays the pet avatar (in this case is a dog), its name, and ages. In the center of this screen, there is a placeholder to put a pie chart

which summarizes the activities of the pet in terms of time. For instance, it is 40% ‘resting‘ and 60% ‘happy‘ in the Figure 4.5. ‘Resting‘ means the pet is idle while ‘happy‘ means it is in motion.

Figure 4.5 Start screen of the iOS application

In the user interface, all the texts in white are actionable buttons. There are seven of them in total. When the button *today* is pressed, it leads to a second screen (see Figure 4.6).

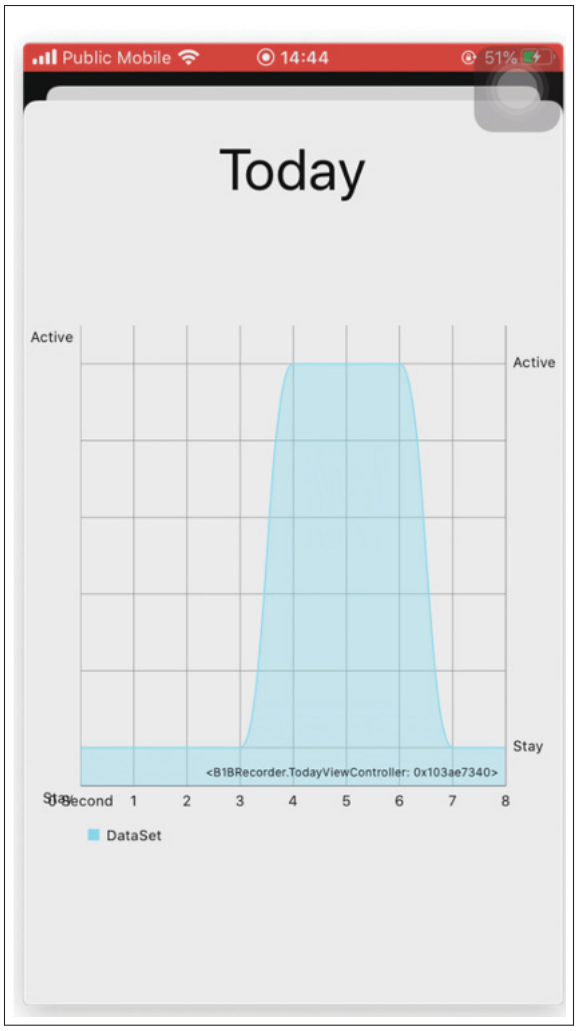

Figure 4.6 Pet activities in a short period

If the *week* button is pressed, the third screen is shown as seen in Figure 4.7. Other buttons such as *start, stop, download, erase, and upload* has a non-user-interface function. Their roles are discussed in the next session.

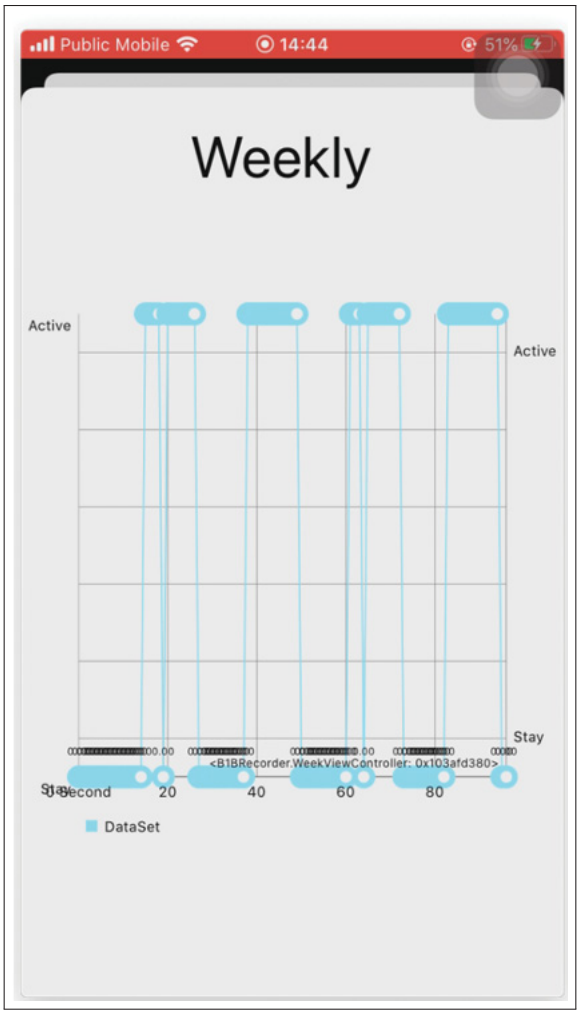

Figure 4.7 Pet activities in a long period

In the second screen (Figure 4.6), the pet activities in time is presented in a filled line graph. The upper bound is *active* (i.e., moving) while the lower bound is *staying* (i.e., idle). The horizontal axis indicates the time points (can be customized to show in second, minute, or hours).

The third screen (Figure 4.7) shows the summary of the pet activities in a week, month, or all time (in the figure, it is from the first moment when the sensor data is record up to the last time when the sensor data is uploaded to a database). The graph has the same legends and the meaning of the axis are the same as those of the 2nd screen.

4.3.3 Functionalities

In the main application screen 4.5, there are 7 buttons (*start, stop, download, erase, upload*) and one label (*download progress*). Here, we make the controlling function explicit via buttons on the screen for demonstrative purposes.

Their functionalities can be categorized into three groups. The first group is to control the collar (*start, stop, download, erase, upload, downloading progress*) while the second group is for behavior prediction (*today, weeks*). The last group (*upload*) is for data storage in a database.

- **Collar controller buttons**:These buttons (Figure 4.8) are inside the main application screen (see Figure 4.5).
  - The start button: turns on sensors inside the collar. When the collar is power on, it will not activate the sensors to collect data until this button is pressed. After it is enacted by a user, sensor data is started recording and written into the collar's internal flash memory. It is worth to note that sensors consume a relatively large amount of energy to run. That's why the current collar prototype cannot operate more than 1 hour while sensors are on but can last for days if they are off.
  - The stop button: turns off the sensors. The collar is still running.
  - The download button: The sensor data is inside the collar's memory storage by default. To make it available to further tasks, these data must be taken out. This button role is to copy all recently recorded sensor data from the collar to mobile application space (i.e., inside the mobile phone internal storage). The data are transferred as is and without any modification. The text *download progress* is activated while data is downloaded by showing information about the ratio of the current downloaded data over total available data.
  - The erase button: because the collar has limited memory capacity, it is better to delete old sensor data after they are downloaded to the phone. This function basically deletes all recorded sensor data inside the collar.

Figure 4.8 Buttons to control the collar

- **Download and upload buttons**: When button *download* is pressed, sensor data is copied into the phone's internal memory. However, the data are in raw format. Its structure is described in Appendix III. This data format is not transferable because it is difficult for other software to read or interpret them. Therefore, in order to increase data portability between software components, they are converted to tabular format in memory. This data then is sent to the in-application prediction part for further processing.

  In this project, we set up and use an instance of PostgreSQL via Amazon Relational Database Service (RDS)[5] Free Tier. By using this service, we can benefit from default networking and security settings as well as scalability if needed. PostgreSQL is selected for this purpose because it is mature and widely used in the industry[6]. Thanks to the variety of extensions in PostgreSQL, it also supports storing time-series data via extensions (e.g., TimeScaleDB [7]) which is suitable for storing sensor data. Also, it is possible to manipulate data entry via standard HTTP commands (RestFul). Their features are useful to extend the project in the future. For a comparison between many database systems, one can refer to Table 4.2.

  When button *upload* is triggered, the sensor data is converted into a tabular format then insert into the PostgreSQL database which is hosted by Amazon. This process requires a constant internet connection.

  Inside the database, we create a database named *b1b* with one table called *sensor_data*. The table contains basic information for data analysis and its schema can be seen in Figure 4.9. In the table structure, user_id is to identify the collar, *time_create* is the moment that sensor data is recorded, while *timestamp* is that moment presented in UNIX time format. *ax, ay, az*

[5] aws.amazon.com/rds

[6] https://www.postgresql.org

[7] https://www.timescale.com

are values from the accelerometer sensor to indicate the position and speed in the 3D space of the collar. An example of data values can be seen in Figure 4.10.

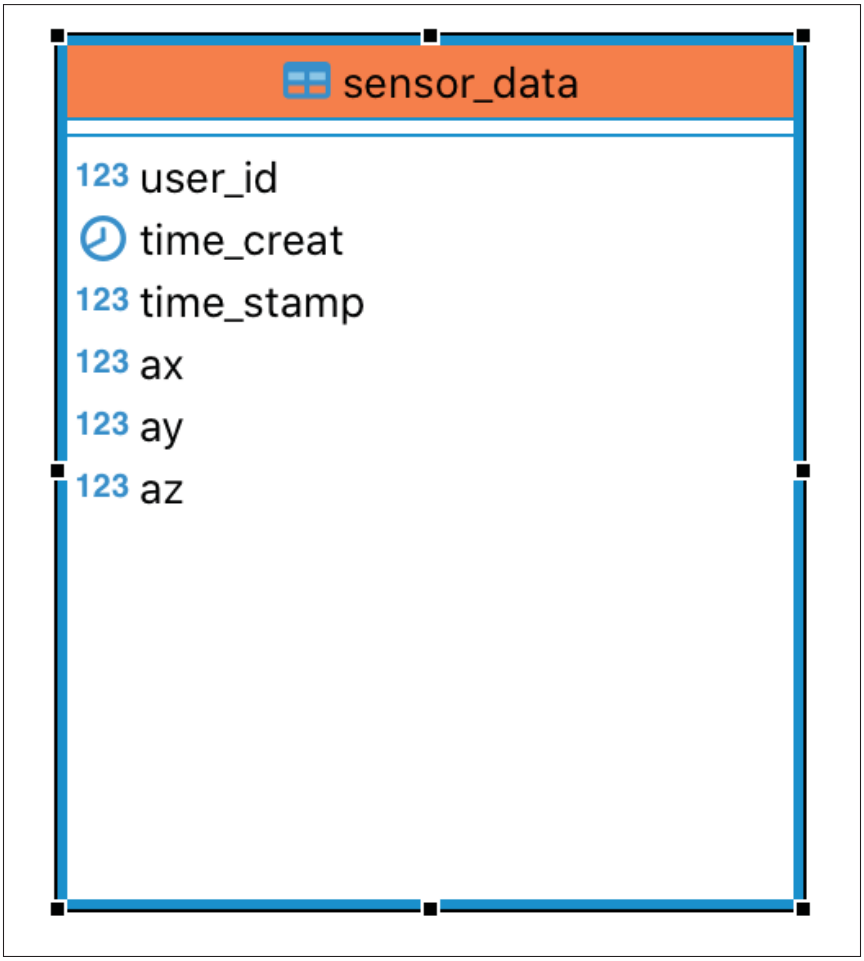

Figure 4.9 SQL schema of the sensor data table

Table 4.2 Comparison of database systems

	MySQL	**PostgresQL**	**SQLite**
First release	1995	1996	2000
Available as cloud service	Yes	YES	No
Network database	Yes	Yes	No
Designed for time series	No	Yes (via extensions)	No
Support Restful	No	Yes (via extensions)	No

sensor_data | Enter a SQL expression to filter results (use Ctrl+Space)

Grid | Text

	user_id	time_creat	time_stamp	ax	ay	az
1	1	2020-04-09 14:12:24	284,243,774	-7,424	-928	1,312
2	1	2020-04-09 14:12:24	284,263,763	-7,408	-880	1,248
3	1	2020-04-09 14:12:24	284,283,752	-7,424	-944	1,296
4	1	2020-04-09 14:12:24	284,303,741	-7,440	-896	1,328
5	1	2020-04-09 14:12:24	284,323,730	-7,376	-928	1,360
6	1	2020-04-09 14:12:24	284,343,719	-7,456	-928	1,344
7	1	2020-04-09 14:12:24	284,363,708	-7,472	-944	1,280
8	1	2020-04-09 14:12:24	284,383,697	-7,408	-960	1,328
9	1	2020-04-09 14:12:24	284,403,686	-7,424	-944	1,328
10	1	2020-04-09 14:12:24	284,423,675	-7,408	-960	1,360
11	1	2020-04-09 14:12:24	284,443,664	-7,424	-944	1,360
12	1	2020-04-09 14:12:24	284,463,653	-7,424	-880	1,296
13	1	2020-04-09 14:12:24	284,483,642	-7,440	-912	1,296

Figure 4.10 An example of table “sensor_data“ content

4.3.4 Loading machine learning model into the application

This step is required for the prediction tasks of the mobile application. The model is loaded from an external source, it is introduced to the project by dragging a model file (.coreml file) into the XCode interface. Xcode then automatically generates Swift model classes based on the coreml file (see Figure 4.11 and 4.12). This model file is the results of analysis part which is described in Section 4.4

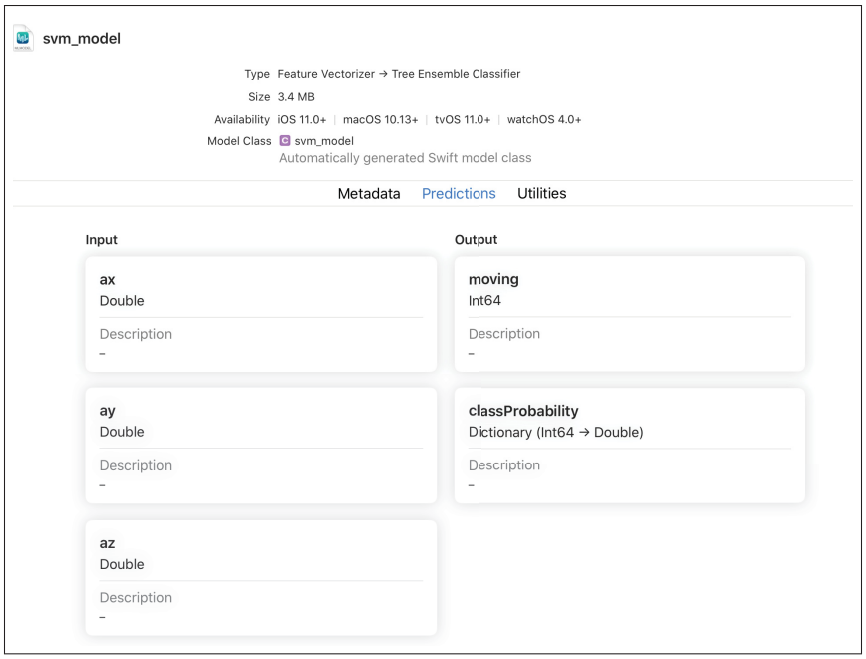

Figure 4.11 A model file presented inside Xcode

```swift
//
// svm_model.swift
//
// This file was automatically generated and should not be edited.
//

import CoreML


/// Model Prediction Input Type
@available(macOS 10.13, iOS 11.0, tvOS 11.0, watchOS 4.0, *)
class svm_modelInput : MLFeatureProvider {

    /// ax as double value
    var ax: Double

    /// ay as double value
    var ay: Double

    /// az as double value
    var az: Double

    var featureNames: Set<String> {
        get {
            return ["ax", "ay", "az"]
        }
    }
```

Figure 4.12 Partial code generated by Xcode for the coreml model file

1. **In-application prediction**: One of the main functionalities of the phone application is to predict a pet behavior by using collected sensor data. It is useful to the owner to track the well-being of their pets within an intermediate time frame like hours or minutes. In order to do that, the application must have the ability to apply the learnt model to these data. In terms of computing power, it is feasible to do because the collected data are relatively small. By doing experiments, we find out that this process does not affect the application responsiveness.

   In detail, the machine learning model is loaded into the application in development time. Then, it is used to interpret the motion of the pet based on the input sensor data. The final result is a two-dimension matrix, which has this form $[time-stamp, predicted-behavior]$. In the end, this matrix is represented in the visual graph as can be seen in Figure 4.6
2. **In-cloud prediction**: For the functionality in the third screen (Figure 4.7), the application downloads prediction results over the Internet and sketches the graph. It is expected that the owner is interested in the all-time well-being of their pet. Therefore, the learnt model is applied in all available data (this can be customized to be within weeks or months). The produced results are then put in the cloud so the application can retrieve and display on the device screen.

   The reasoning behind these tasks is based on the assumption that the more data are available, the more computing resources are needed. It could affect the stability of the phone or the application responsiveness (e.g., the phone does not have enough space to store all sensor data).

4.4 Data analysis

In this section, we discuss the data analysis procedures. We use the two datasets for demonstration purposes. The first one is BeOneBreed dataset which is provided by our industrial partner. The second one is an academic and public dataset named Human Activity Recognition Using Smartphones (HAR) (Anguita, Ghio, Oneto, Parra & Reyes-Ortiz, 2013). However, we focus more on the BeOneBreed dataset as it is gathered from the prototype collar.

4.4.1 Dataset

1. **BeOneBreed's dataset**: The dataset was acquired under the responsibility of BeOneBreed and kindly made available to us for this research study. It is labeled at industrial partner facilities. There is a controlled environment (e.g., room) with video cameras, a pet with a collar, and an experimenter. The experimenter annotates the behavior of the pet based on the time, video and sensor data. Figure 4.13 is an illustration of the control environment setting.

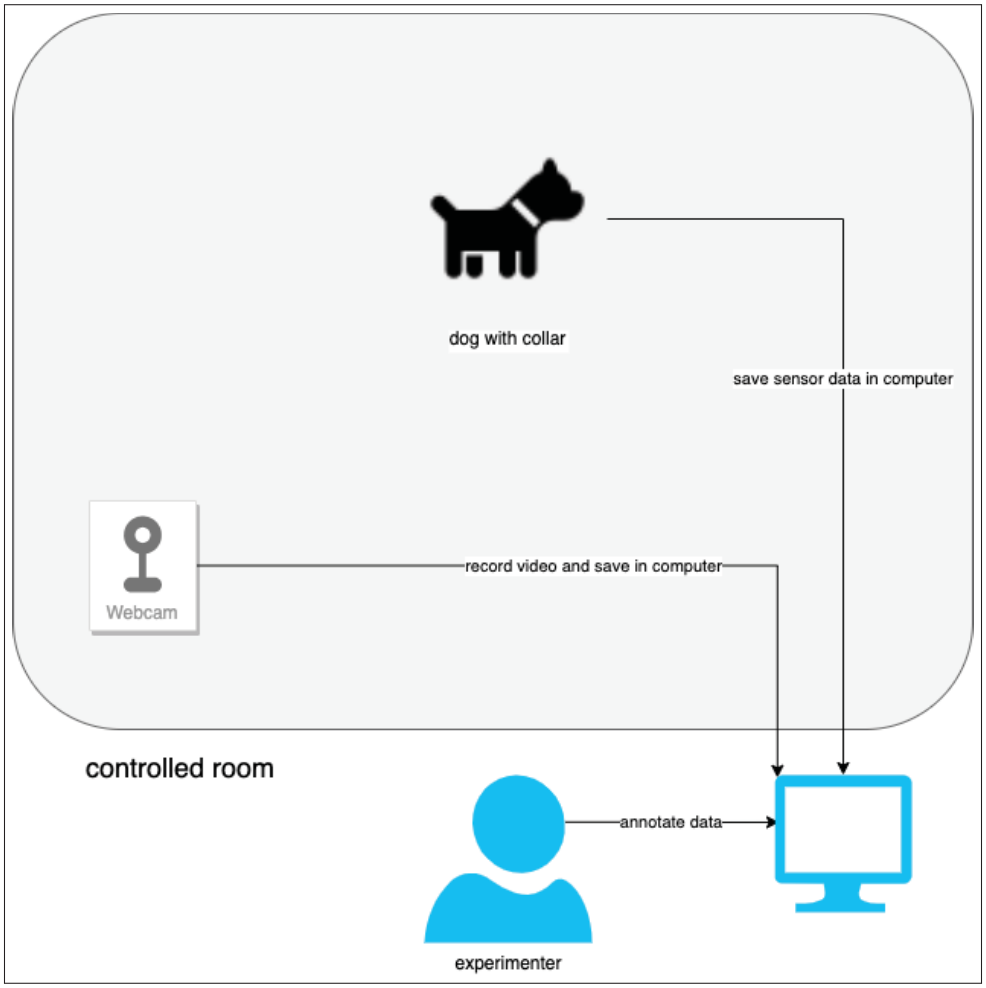

Figure 4.13 Settings of the controlled environment to record the dataset

Those data are recorded and annotated in four sessions spanning three different days.

The annotated data contains 499,434 samples and are in csv format. With the sampling rates of 50Hz/second, that is around 3 hours of training data. Table 4.3 shows an example of 10 selected values (gravitational acceleration g).

Table 4.3 10 selected values of training data

index	**ax**	**ay**	**az**	**label**
0	-0.6688	-0.1520	-0.2368	0
1	-0.6720	-0.1392	-0.2336	0
2	-0.6592	-0.1312	-0.2224	0
3	-0.6384	-0.1216	-0.1904	0
4	-0.6432	-0.1120	-0.1648	0
5	-0.1104	-0.2240	-0.4944	1
6	-0.0960	-0.2480	-0.5456	1
7	-0.0864	-0.2352	-0.6000	1
8	-0.0816	-0.1952	-0.6080	1
9	-0.6464	-0.1680	-0.0736	1

In this data set, ax, ay, and az are accelerometer sensor data and the label is its associated pet action. Label 0 means the pet is stationary while 1 means that it is moving. The folder name is named after the sensor recording date (e.g., folder *06-03-2019* stores data recorded on 06-03-2019) . The overview values of ax, ay, and ax are described in Figure 4.14.

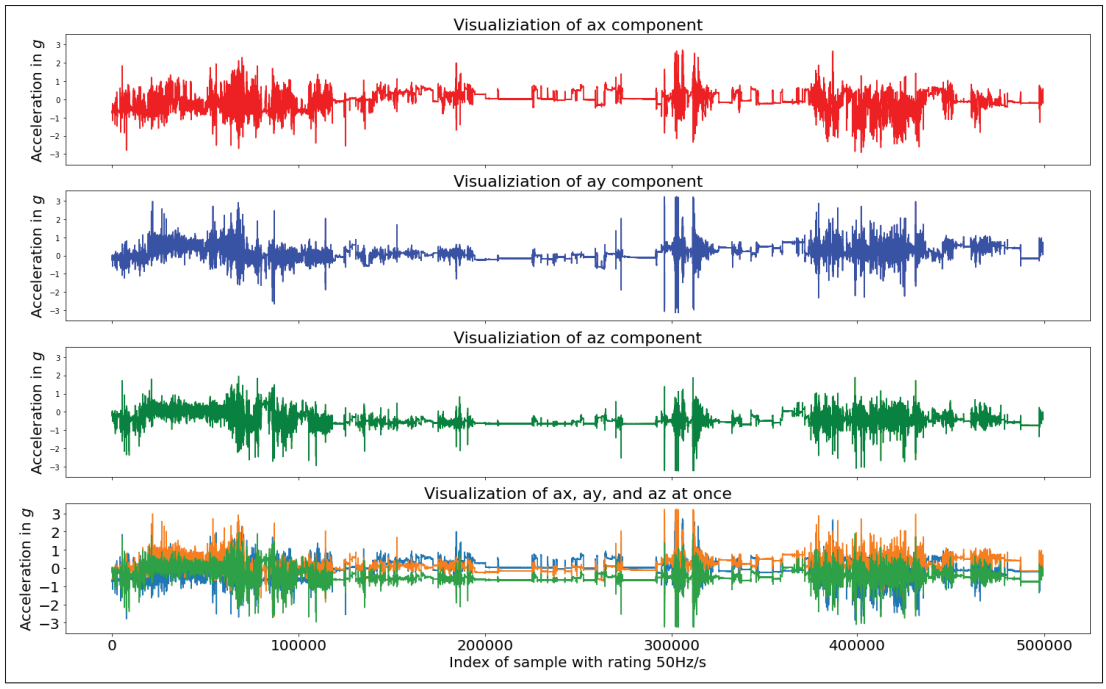

Figure 4.14 Visualization of dataset in separated components

By performing descriptive statistical analysis on the training set, we find out that the data have outliers. This could affect the prediction algorithms, and they should be removed before feeding into a learning procedure. The overview statistics can be seen in Table 4.4.

Table 4.4 Descriptive statistics of the training data

	ax	**ay**	**az**
counts	499,434	499,434	499,434
min	-2.9152	-3.1392	-3.2528
max	2.7104	3.2416	1.9584
label 0 [stationary]	453,999	453,999	453,999
label 1 [moving]	45,435	45,435	45,435

For further analysis, we examine the distribution of each feature in the dataset. The distribution of ax, ay and ax are around the center mean shows that the number of outliers is small (Figure 4.15). This indicates that the data is good for analysis because many outliers can distort many statistical measures when fitting the model (Friedman, Hastie & Tibshirani, 2001).

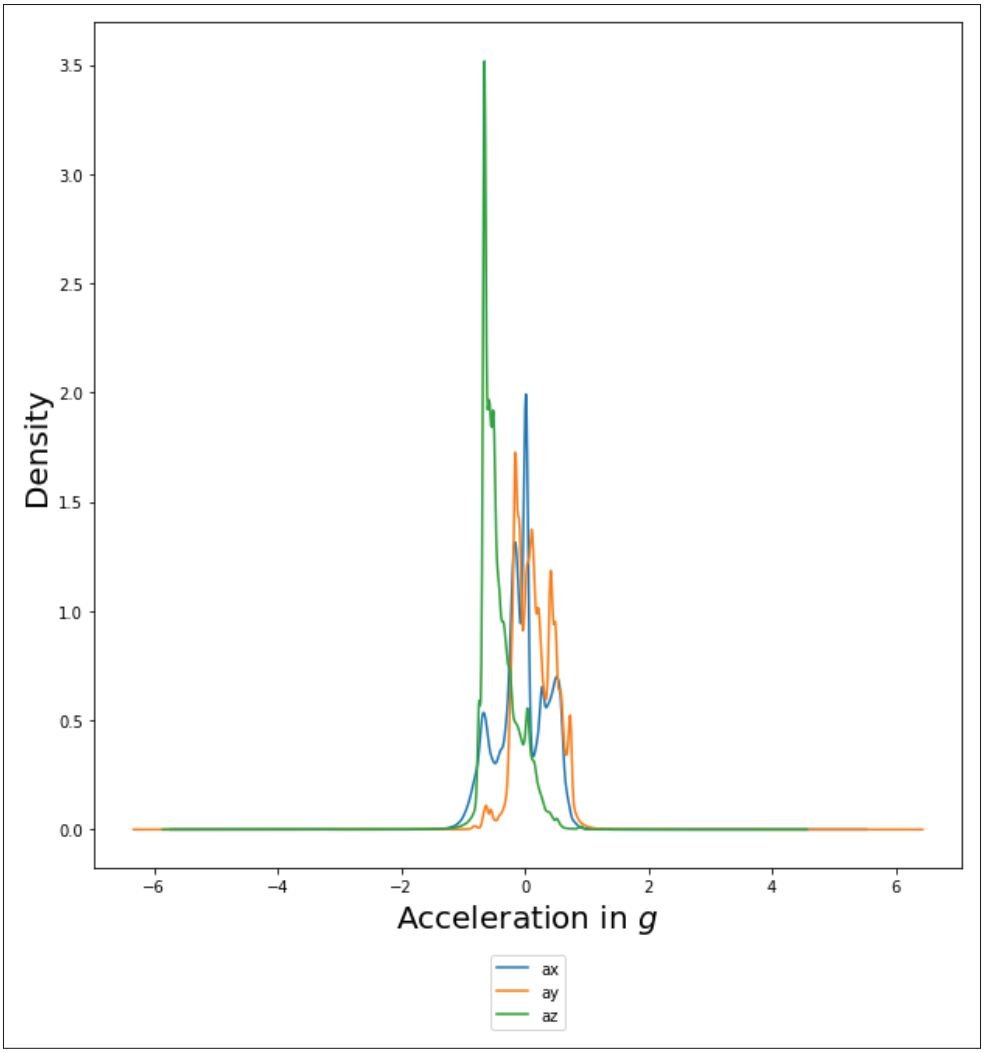

Figure 4.15 Distributions of the training data

Moreover, the training data set is considerably imbalanced (see Table 4.4), with the major data point is in class 0 (i.e., stationary or not moving). That imbalance ratio is more than 10 times between class 0 and class 1 (see Table 4.4). This could cause many problems (He & Garcia, 2009; Chawla, 2009) in measuring the learning performance and behavior of machine learning algorithms (e.g., bias and in favor of the majority class, mis-prediction the minority one). Therefore, the imbalanced problem must be addressed before fitting the model.

Figure 4.16 is a graphical representation of the first 3000 raw data points from the dataset. Intuitively, one can observe that there are few significantly different points - which might be fast ‘moving‘ in this period then following by many fluctuations.

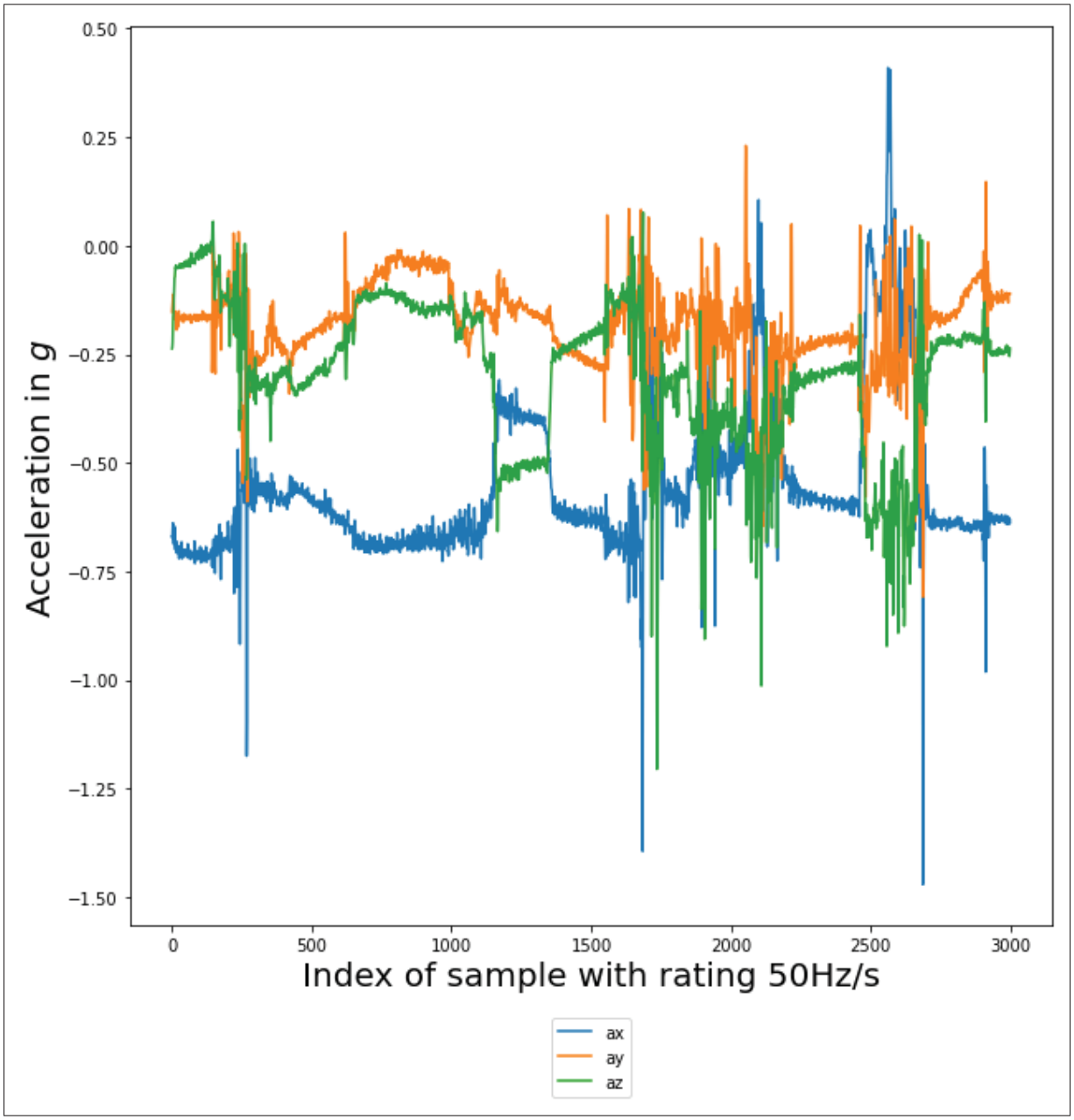

Figure 4.16 Sample data of the first 3000 data point

However, the patterns are not straightforward when looking at the whole dataset. For instance, visualization of ax based on classes (Figure 4.17) shows that the two classes are clearly not separated. It requires further processing to classify the two classes.

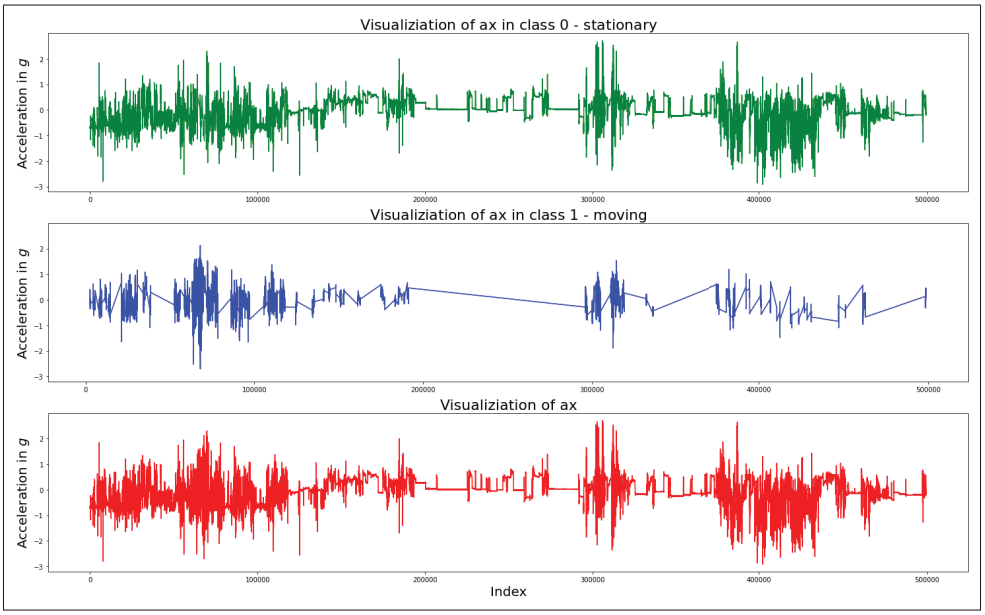

Figure 4.17 Visualization of ax in dataset

2. **Human Activity Recognition Using Smartphones Data Set**: For demonstrating the ability to recognize multiple activities using sensor data, we select the data-set - UCI HAR data-set (Anguita *et al.*, 2013). It is because in the BeOneBreed dataset, it has only two types of motion for the dog (0 for stationary and 1 for moving). Moreover, to the best of our knowledge, there is no similar public dataset about pet motion available.

   The data are collected from 30 persons aged between 19 and 48 performing one of six standard activities while wearing a waist-mounted smartphone. The movements then are annotated manually by associating with their videos. The example of how this task is done can be seen on the video at https://www.youtube.com/watch?v=XOEN9W05_4A. The six activities performed are: walking, walking up-stair, walking downstair, sitting, standing and lying.

The movement data is recorded by ax, ay, az accelerometer data and gyroscopic data at 50Hz. The same rate as of BeOneBreed dataset. In addition, the fixed position of the smartphone on a human waist is comparable with the collar on a pet neck.

However, the raw data is not provided. Instead, the treatment version is publicly available for downloading at https://archive.ics.uci.edu/ml/machine-learning-databases/00240/UCI%20HAR%20Dataset.zip. The treated steps are pre-processing accelerometer and gyroscope using noise filters, splitting data into windows of 2.56 seconds with 50% overlapping, and dividing accelerometer data into gravitational and body motion components.

As results, the treatment data contains 561-featured vectors, scaled in range $[-1, 1]$ and their annotated labels, which are ready for further analysis. Figure 4.18 visualizes the first 100 data point of this data set based on its activities. The data is spitted into 70% for training and 30% for testing.

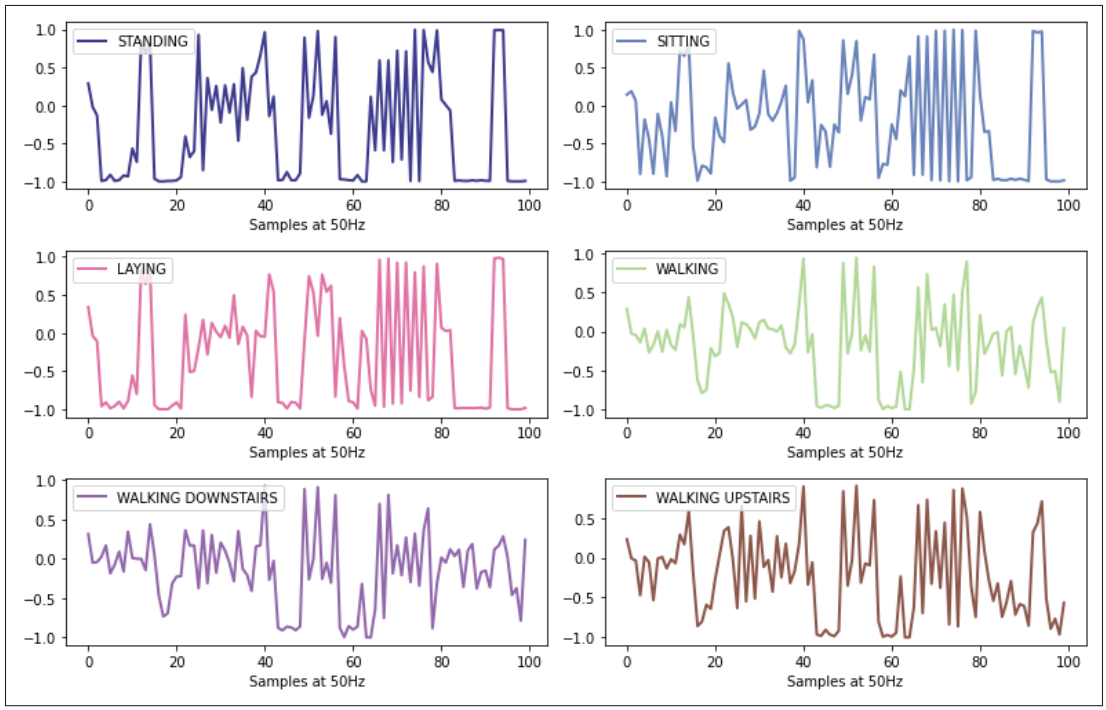

Figure 4.18 Visualizing 100 samples from the HAR dataset

4.4.2 Analysis approach for BeOneBreed's dataset

The thesis aims to analyze sensor data to predict several movement types of pets (such as running, sleeping, walking normally, jumping, etc.). However, due to the limitations in data collection, BeOneBreed dataset records only two types of behavior: moving (i.e., walking normally) and staying. Also, the prototype collar includes only an accelerometer at this research stage.

In preliminary analyses, we calculate the $l1, l2$ and max norms of features (ax, ay, and az) and visualize them according to the label (1 for moving, 0 for stationary) (see Figure 4.19). Given vector x which has n components, $x = (x_1, x_2, .., x_n)$. The $l1$ norm is the sum of absolute values of all components of the vector and defined by the following equation:

$$||x||_1 = \sum_{i=1}^{n} |x_i| \tag{4.1}$$

$l2$ norm is

$$||x||_2 = \sqrt{x_1^2 + x_2^2 + .. + x_n^2} \tag{4.2}$$

and the max norm is

$$||x||_\infty = \max\left(|x_1|, |x_2|, ..., |x_n|\right) \tag{4.3}$$

Also, we use PCA for projecting all features into two principal components (pca-one and pca-two) (see Figure 4.20) to see how they are clustered. These preliminary analyses show clearly that the data are not linearly separable. More advanced analysis is required. In this work, we use machine learning to address the problem.

Furthermore, this work aims to be able to utilize data from multiple sensors (such as gyroscopic, accelerometer, skin temperature, ambient temperature, etc). To that extent, the dimensions of data (could be hundreds) and data structure are much more complex. In such cases, machine learning approaches also prove its popularity amongst studies (see Section 1.3). As a result, to investigate the ability of method, we use standard learning algorithms to predict the pet behavior.

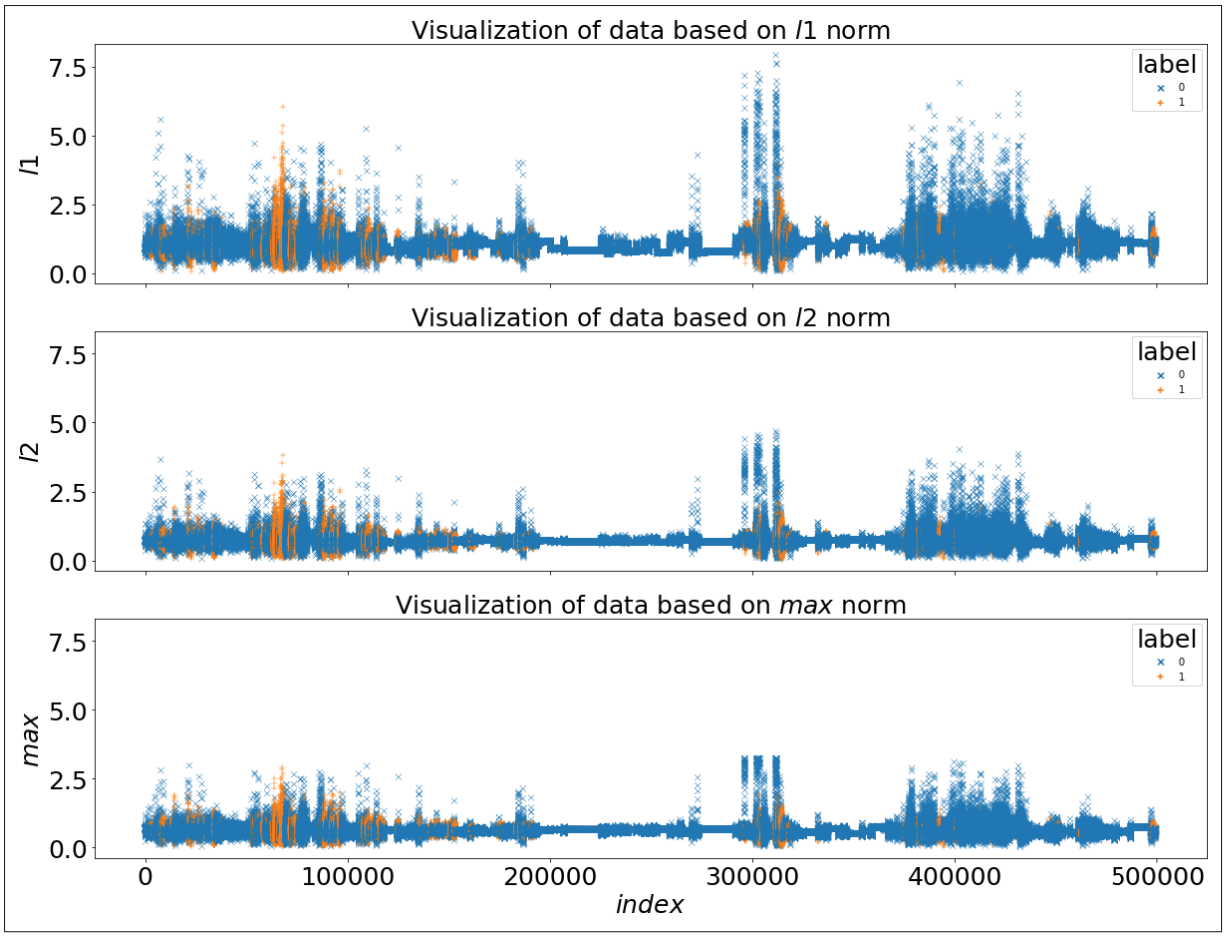

Figure 4.19 Visualization of data based on $l1$ norm

We apply machine learning models on BeOneBreed's dataset to extract insight data. The model which yields the best performance is selected and applies into future data to get the predictions. The overview of this process can be viewed in Figure 4.21. In the next part, we discuss the algorithm selection processes.

4.4.3 Machine learning algorithms

We apply 3 machine learning algorithms to the dataset: logistic regression, random forest classifier and SVM. We select these algorithms because they are widely used in various works (Haladjian *et al.*, 2017; Ladha *et al.*, 2013b; Bayat *et al.*, 2014; Haladjian *et al.*, 2018)

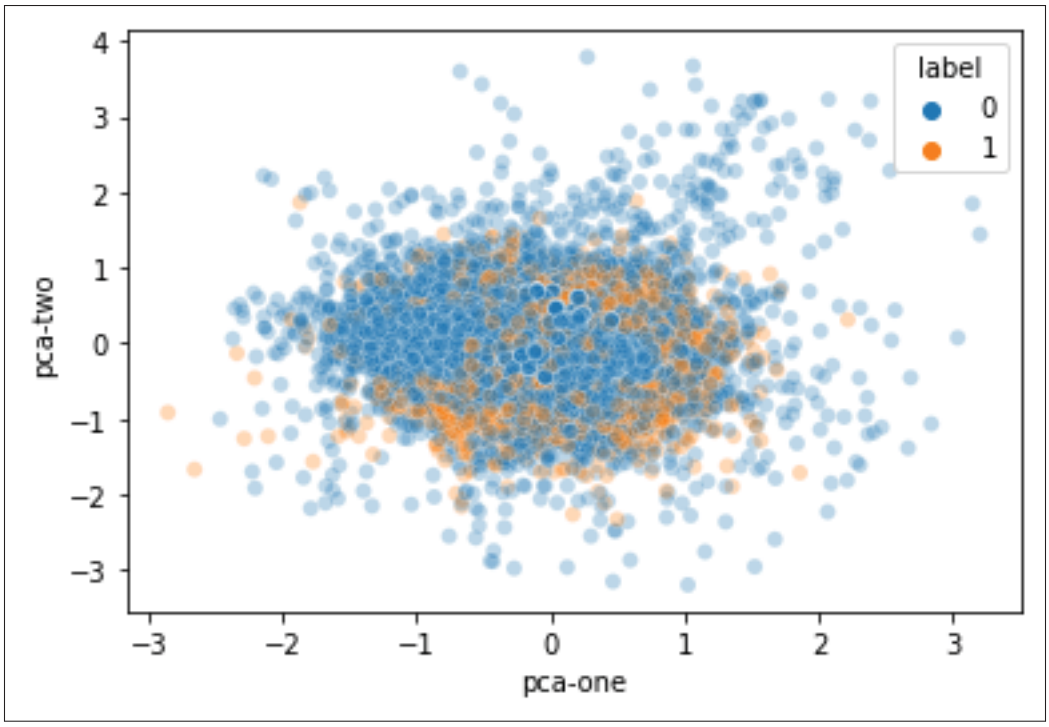

Figure 4.20 Visualization of data based on PCA

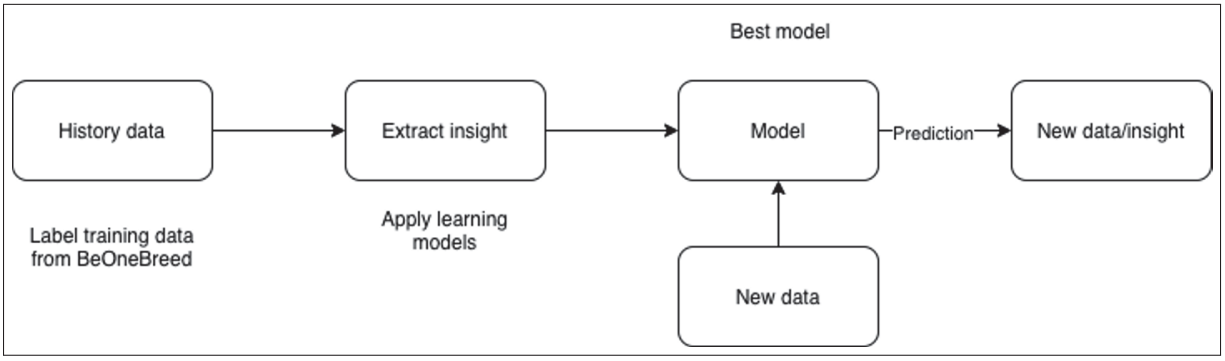

Figure 4.21 Overview of data analysis flow

about animal behavior prediction and they prove to be highly accurate. Although linear regression is relatively faster, it is not used because this is a classification task with two discrete categories.

In addition, we use logistic regression on this data because of its mathematical nature: it maps continuous data into a binary classification: 0 or 1. This fits into the problem because there are only two pet behaviors to forecast in this case. Another advantage of this algorithm is that in

practice, it can be adapted to the multiple class classification problem[8]. This is useful when there are more than two pet behaviors are labeled in the data.

Random forest is an ensemble machine learning algorithm. By combing multiple weak KNN classifiers, it creates a robust, precise one, an example can be seen in (Bayat *et al.*, 2014). This is useful for animal behavior problems because the data is volatile and can contain outliers. However, random forest is extremely sensitive to the class imbalance. So the data must be pre-processed to eliminate those features because applying this algorithm.

The last algorithm in the benchmark is SVM. This is another popular machine learning algorithm uses for classification. SVM is good for our case because it is a memory-efficient technique, which is positive for a mobile device. Another advantage is that SVM can work well with multiple dimension data, which might a problem we need to tackle when several sensor measurements are recorded.

4.4.4 Training data preparation for BeOneBreed's dataset

Based on the preliminary investigation in Section 4.4.1 , it is found out that the data is strongly imbalanced: 453,999 data points are in class 0 (stationary) and 45,435 data points are in class 1 (moving).

Because of this significant imbalance, the data must be re-balanced before feeding into a machine learning algorithm. If this imbalanced data-set is used for machine learning procedures, it has a negative impact on model fitting (Friedman *et al.*, 2001). One technique for resolving such a problem is to sub-sample. There are two sub-sample methods:

- Up-sampling: randomly sampling with replacement to make the minority class to be the same size as the majority class.
- Down-sampling: randomly sub-setting the majority class in the training set so that its class frequencies match those of the minority class.

[8] https://scikit-learn.org/stable/modules/multiclass.html

For this behavior recognition task, the experiments show that up-sampling creates negative results on model fitting while the down-sampling makes it positive. Therefore, we choose the down-sampling technique. Additionally, we split the data in a ratio of 80/20 (Sanders, 1987). 80% of data is used for training tasks and 20% is for testing.

4.4.5 Model selection for BeOneBreed's dataset

After data preparation in Section 4.4.4, we perform model training process to find the best parameters for each algorithm. After that, we apply the model to the remaining 20% of testing set to predict their classes. By comparing the predicted pet behaviors with actual pet behaviors of all data samples, we can evaluate its performance by calculating the F1-score (Chinchor, 1992). This F1-score is used to compare performance between different algorithms, which is higher is better. In other words, we choose the algorithm which provides the highest value of the F1-score as the final model. That process is summarized in Figure 4.22.

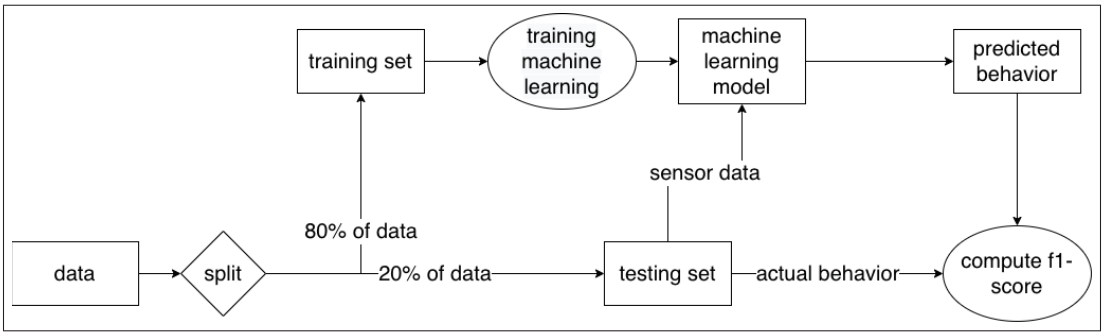

Figure 4.22 Flow chart to evaluate performance of a machine learning model

Table 4.5 describes the F1-score measurement of Logistic Regression, SVM and Random Forest classifier after we apply them on the dataset. It is clear that Random Forest outperforms the other approaches in this task with an obtained F1-score > 0.8 for 2-class classification.

The model is then saved in CoreML format in computer hard disk. It is later transferred into an iOS application for forecasting tasks. The whole process in Section 4.4.1 and 4.4.4 is described in Figure 4.23.

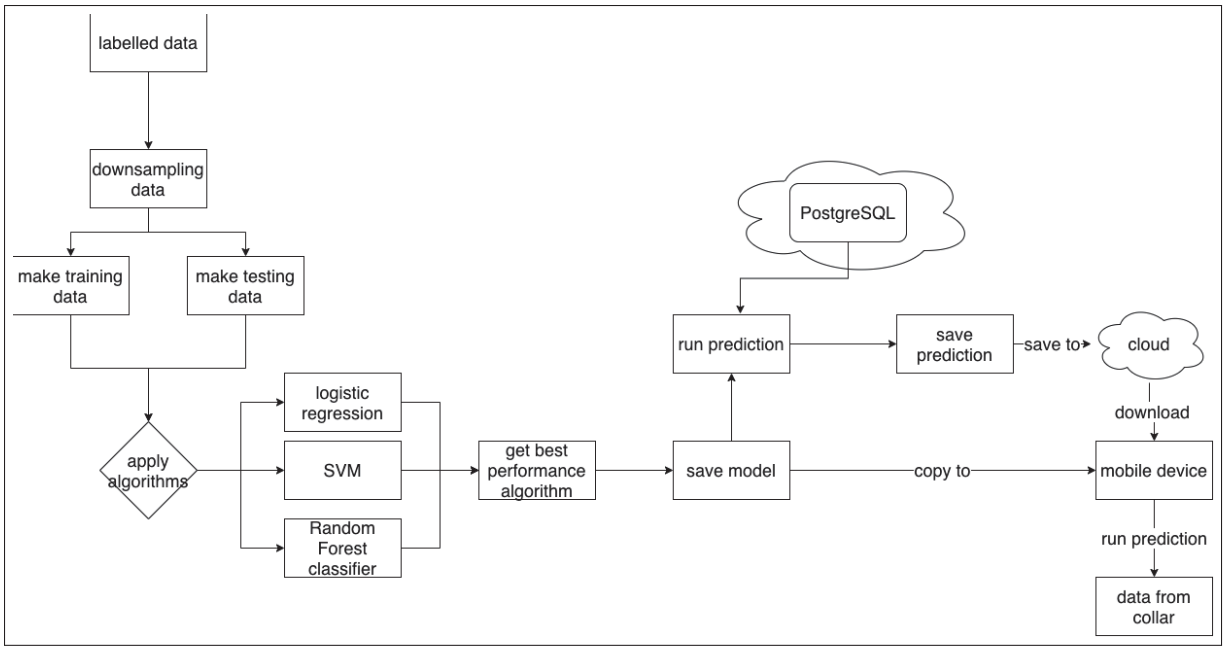

Figure 4.23 Overview of the machine learning model selection pipeline

Table 4.5 F1-score

	f1-score (class 0)	**f1-score (class 1)**
Logistic Regression	0.52	0.60
SVM	0.73	0.74
Random Forest	0.81	0.81

There are various tools that are utilized in this step. For data preprocessing, Pandas[9] (Wes McKinney, 2010) is used. For machine learning model evaluation, we choose scikit-learn [10] (Pedregosa et al., 2011). The scikit-learn model is converted to CoreML format by coremltools[11] which is developed by Apple Inc. The whole process is executed inside Jupyter[12] notebook and used Python programming language. The whole procedure is run on a dedicated computer running a Linux-like operating system.

[9] http://pandas.pydata.org/

[10] https://scikit-learn.org

[11] https://apple.github.io/coremltools/

[12] https://jupyter.org

4.4.6 Model usage

The result of previous processes (Section 4.4.5) is a model that is used to predict the pet behaviors for both in-application and in-cloud.

1. **In-application prediction**: The model is integrated into the mobile application and applied to newly downloaded data from the collar. In this approach, the performance of prediction tasks is depended on processing power of the iPhone device.
2. **In-cloud prediction**: In this method, we apply the model to all available data inside the database. First, an application script is created to query all data from PostgreSQL. After that, each data point is fed into the model. The prediction results are then saved as an external file which can be accessed via a URL (Universal Resource Locator) on the Internet. For this project, we put the results in a CSV file and it is located in github.com. Later, the mobile application retrieves this file and visualizes the data in the graphs (see Figure 4.7).

4.4.7 Human Activity Recognition Using Smartphones dataset analysis

In order to demonstrate the effectiveness of machine learning method in behavior prediction, we apply the same sets of algorithms (SVM, Logistic Regression and Random Forest Classifier) on the UIC HAR dataset as described in Section 2.. In this dataset, the data is cleaned and no further pre-processing is needed. The train and test sets are divided by the authors. There are 7352 records for training and 2947 for testing. The size of feature vector is 561 and there are six classes which match with six human activities.

Table 4.6 F1-score results for each class

	class 1	**class 2**	**class 3**	**class 4**	**class 5**	**class 6**
Random Forest	0.93	0.89	0.91	0.90	0.91	1.00
Logistic Regression	0.97	0.95	0.97	0.92	0.93	1.00
SVM	0.96	0.94	0.95	0.91	0.93	1.00

Class 6 which is associated with the action of “lying“ is always correctly predicted likely because this type of activity triggers the least fluctuation in sensor data values.

The results here demonstrate that it is possible to recognize multiple pet behaviors by adding more sensors to the research system or collecting more behavior data.

4.5 Limitations

Before moving to the development of the demonstrator, ultimately developing commercial systems, there are certain limits that need further investigation when implementing the system. The following limitations are amongst the most important:

1. **Power consumption**: energy consumption of the collar and the mobile devices are not measured yet.
2. **Hardware**: the collar has only one sensor that is an accelerometer. It is beneficial to have more sensors to potentially identify finer behaviors (e.g., vital signature sensors). The prototype collar battery life is short (about 1 hour). Mechanical design was not of a concern at this stage. For example, the casing was not designed to resist to actual long-term usages of dogs.
3. **Software**:
    a. The in-collar behavior recognition software part is not implemented.
    b. The mobile application supports only one collar and one pet owner.
    c. The long-term analysis at the cloud part is executed manually.
    d. The database schema is for a single device.
    e. The mobile application is only for iOS devices.
4. **Data**: there is a small amount of training data available which covers for a few hours. The data have only two labels: stationary and moving. As a result, the machine learning model can only provide the prediction for two kinds of activities of the pets.

4.6 Chapter summary

In this chapter, we covered the implementation of the demonstrated system. That implementation consists of a mobile application, the data storage strategy, and data analysis procedures. Particularly, the mobile application user interface and functions are described in detail. Sensor

data are temporarily stored inside the collar and the mobile application before transferring into a database system in the cloud. The data analysis is executed at the mobile application for immediate results. For larger data, it is performed at a remote computer and the results are sent back to the mobile application for visualization. Also, we discuss how the data analysis is implemented and its evaluations.

Although there are several limitations in this system, we showed many possibilities and the potential of the developed system. The system is feasible to solve the data storage challenge. In addition, the system demonstrates the usage of BLE for communication at low-power devices (e.g., the phone and the collar). The system also exposes that it is effective to utilize machine learning algorithms to recognize pet behaviors. Based on the development criteria defined in Section 3.4.5, the system successfully completes all of them.

CONCLUSION AND RECOMMENDATIONS

In this research, we aimed to investigate the possibilities of smart toy system for pets. In order to do that, we proposed potential system architectures and functions of their components. These architectures have the potential to solve the most two crucial challenges in developing that are data storage and energy consumption.

In summary, we proposed the following key points:

1. **System architecture**: we proposed multiple possible system architectures for realization of a smart toy system for pets. These system architectures are to tackle the two most crucial problems: data storage and energy consumption. We then analyze the pros and cons of each design associated with their technical choices. As a result, we recommended the most viable solutions to implement the smart toy system for pets. By doing this, we provided the answers to research questions (Q1 in chapter Introduction). The answer is "yes": it is possible to design architectures for smart toy systems.
2. **Demonstration system**: we proposed and developed a smart system demonstrator for pets based on one of the most viable architectures, including the use of ML techniques for data analysis. As a result, we answered that it is possible to implement a smart toy system architecture (research question Q2) and it is also to provide useful information based on historical data of pet behaviours (research question Q3).

Although there are several architectures, their general components are similar. All the system consists of a mobile application, a collar, toys, other connected devices, a database, and sensor data analysis applications. The smart toy system uses major wireless technologies (e.g., BLE or WiFi) to communicate between each device and WiFi to access the Internet.

However, it is hard to validate the feasibility of each system architecture due to its high abstraction level. Therefore, we proposed a smart system demonstrator for pets based on one of the most

viable architectures. This demonstrated system consists only the most critical parts: a collar, a mobile application, and cloud parts.

- **The wearable collar** has an accelerometer sensor and an integrated battery. It connects to the mobile phone via BLE.
- **The mobile application** can control the collar such as turning on/off, downloading sensor data, and erasing the collar's flash memory. It can also do on-device data analysis for temporary data. Additionally, it downloads the predicted results from the cloud resources to show on the dashboards.
- **Sensor data** can be stored ephemerally inside the collar and then are downloaded to the mobile application for further analyzing. After that, the data is converted into a tabular format before being uploaded to a RDMS (PostgreSQL) which is hosted on Amazon cloud services. As results, data storage limits do not depend on the internal memory capacity of local devices.
- **Data analyses** are executed inside the mobile application and in the cloud. For small and temporary data, it is analyzed inside the mobile program. For permanent data in the cloud, we use a remote computer to perform the analysis. By doing this way, we limit the energy consumption of the local device by off-loading the computing intensive tasks to the appropriated devices.

In summary, we point out that it is possible to use our design to create a smart toy system for pets. Although we have implemented the demonstrated system, there are certain limitations that should be addressed in the future.

1. **System implementation**: the whole system design is not fully implemented. It is missing the IoT hub, connected devices and smart toys. The toys and the hub are still in the design procedures and, therefore cannot be tested its interaction within the whole system.

2. **Data analysis**: the data analysis has limits in training data and modelling algorithms. Currently, its training set has only two types of action: stationary or moving. As a result, only stationary or moving behavior can be recognized by the software.
3. **Collar**: the hardware is still an early prototype stage and has limits. It has only the accelerometer sensor. Also, the device has enough energy to power the sensor for about 1 hour.
4. **Mobile application and database**: the current implementation is designed for a single user with a collar device and for iOS devices. The design can be adapted in the future by modifying the table schema in PostgreSQL and adding multiple user management modules in the iOS application. In addition, the current mobile application is only for Apple iOS-compatible devices.

According to the previous limits, we recommend certain tasks should be enhanced in the future. They are the demonstration system design, the data analysis software, the collar, and other devices.

1. **Demonstration system**: this system should be designed to include other toys, connected devices and a hub. It should define the interaction protocol between those devices. Also, it should be created in a way that energy consummation is possibly recorded.
2. **Data analysis software**: this part should be improved. Future researchers can investigate to use special time series analysis algorithms such as LSTM (Long Short Term Memory) or RNN (Recurrent Neural Network). At the same time, the data must be collected to get additional pet behaviors.
3. **Collar**: the collar functionalities could be enhanced by adding more sensors (such as magnetic field, ambient temperature). Such sensors could be used to detect nearby toys and objects, or collect pet's vital data. Simultaneously, its firmware must add new functionalities associated with the newly installed hardware. Moreover, the battery should be improved by

testing or redesign. For validation purpose, the collar firmware should have the abilities to measure power consumption when performing certain tasks.

4. **Smart toys and IoT hub**: should be developed to work with the collar. The toys should be added because it will significantly increase the usability of the system. The hub can act as the local processing unit to analyze and synchronize data with the other parts over the Internet.
5. **Software**: the mobile application and back-end database in PostgreSQL should add features to allow multiple users to operate at the same time. The data analysis part should be improved by collecting more labeled training data. The mobile application should be extended with more functions such as alerting abnormal pet behavior.

APPENDIX I

CONFERENCE PAPER

Proceedings of the Canadian Society for Mechanical Engineering International Congress 2020
CSME Congress 2020
June 21-24, 2020, Charlottetown, PE, Canada

Towards a smart toy ecosystem for pets

Van-Tien Hoang, Lucas Hof
Department of Mechanical Engineering
École de technologie supérieure
Quebec, Canada
van-tien.hoang.1@ens.etsmtl.ca
lucas.hof@etsmtl.ca

***Abstract*—When pet's owners take care of their pets, it sparks the need to know about the well-being of their pets. This requires the ability to monitor the animals' activities in order to recognize unexpected behavior of the pets. Such monitoring systems are also useful for pet training purposes, which could be extended to more intelligent systems used to encourage specific behavior of pets. In order to do so, this study aims to develop a system that can keep track and record of such data of pets (e.g. movements, body temperature). The proposed eco-system includes a collar affixed to the pet, interacting with a mobile application, a cloud data storage and processing platform as well as other toys to communicate with. This system is designed to be easily extended with new toys. It also can interact with smart voice assistant systems such as Apple Siri or Amazon Alexa. In addition, it has light weight so it can be affixed to small dogs or cats.**

***Keywords*-Internet-of-Things (IoT); software architecture; pets; smart systems; pet behavior classification; mobile applications; cloud computing**

I. Introduction

Human has started living with pets for thousands of years [1]. That long history of relationship between human and pets create strong bonds between them ([2, 3]). That could be one of the reasons why there are more than 400 million domestic dogs world wide [4]. Particularly, in the United States of America, there are more than 180 million dogs and cats in 2017 [5].This results in a huge expenditure for pets, e.g. more than 69 billion dollars spending on foods, medical and other care (see more [6]). In particularly, the market sales of smart pet products is $565 million in 2018, which is up 11% from 2017 [7].

When the owners take care of their pets, it leads to the need to know about their well-being. There is an ongoing trend that pet-owners want systems to monitor their pet's activities and to recognize their behavior patterns. These system must be able to detect abnormal activities to alert and support the owner to take corrective measures [8, 9].

A. Review of related work and products

Currently, there are several works related to smart-systems for monitoring and behavior recognition. These can be categorized as commercial products: 1) available in the market and 2) in test or research phases.

1) Commercial products: An example of a commercialized product is Mookkie [10], a smart bowl identifying each pet visually ("face ID") and associate them with specific food. As a result, the diet can be customized for each pet. The system comes with a dedicate smartphone application providing notifications and short videos to the owners. However, this smart bowl does not interact with other "Mookkie" bowls or with other connected devices in the house.

Another product that helps the owner to achieve optimal pet diets is Sure Petcare's Pet Feeder Connect [11]. This product uses a scale to create a specific portion of food for pets. It also monitors the pets' food consumption behavior. This system identifies each pet using a wearable tag for each animal. The feeder connects to a 'hub' via a low energy protocol. The hub then uses a WIFI connection to communicate with the mobile phone, which in turn connects to the internet.

Petpace [12] is also a product to monitor the pets' well-being. It consists of a collar monitoring dog's vital data and alerts the owner when abnormality happens. Its architecture consists of three layers. The first layer contains the smart sensors affixed to the dog's collar. This sensor sends the actual data to the middle layer which is a base station. The base station communicates with the sensors and combines the data to send them to the cloud layer using the standard Ethernet interface. The calculation and analysis happen at the cloud layer. The owner can accesses those data wirelessly using their mobile application to consult their dog's Health Index, Activity Index, and Burned Calories.

Similar to Petpace, Fit bark [13] is a dog activity and sleep monitor device. It uses Bluetooth Low Energy (BLE) to make the connection with a WiFi base station. This technology promises lower energy consumption during data transmission, extending the lifetime of the battery in the collar. Any Android or iOS operated smartphone can be used as base station.

Another device, Scollar [14], is a dog tracking solution that provides information about the dog's activity and which offers tracking functions. This system contains a base station (e.g. a smartphone) including a WiFi connection. Data collection and transfer is the same as in the previous devices. The main difference is that Scollar is the only device that provides a GPS sensor and dog tracking service. The collar also contains a temperature sensor and integrated battery, with an estimated lifetime of 45 up to 60 days.

2) Research studies: In addition to these commercialized products, there are multiple academic studies about wearable devices for animals. For example, Pet Buddy [15] is a device for dogs which aims to monitor physical behavior

of the pets. The device is based on an Arduino mainboard, including gyroscope sensors. The collected sensor data are later transferred via Bluetooth to another Arduino-based logic board, which itself is connected to a computer to analyze the canine behaviors. Similar to Pet Buddy, WagTag [16] is another solution to track dogs' activities which uses Bluetooth to transfer data to the computer.

Instead of monitoring basic physical activities, David Sec et. al [17] propose a system to monitor health condition of dogs to detect possible epileptic seizures. This wearable device has a noise sensor to identify respiratory frequency and it includes a heart beat sensor. It exchanges data via BLE 4.0 with a Raspberry running a Windows IoT operating system. The collected data are sent to a processing server via file transfer protocol (FTP).

The similarities of the three aforementioned works are the isolation of the wearable devices: they are not designed to be integrated with external services or operate in a network with other connected IoT devices.

One example which demonstrates the possible interactions between multiple devices is from Y. Guo et. al [18]. They prototype a wireless sensor platform for livestock (i.e., cows) behavior tracking. The board has a GPS receiver and temperature, magnetometer, and accelerometer sensors. The nearby animals of the herd form a wireless sensor network to relay the data to the central server. However, due to the integration of multiple sensors and functionalities, the device is too heavy for domestic pets. Also, domestic pets are not constantly near each other to form such a wireless sensor network.

3) Related products designed for humans: Besides aforementioned solutions, there are other devices close to the project. These products inspire us to develop the collar and the mobile application's functionalities.

- Oblu [19]. Oblu is a universal sensing platform for IoT using BLE 4.1 for communication. This development kit contains sensors such as an accelerometer, a gyroscope, and GPS, and it is a potential candidate to build our wearable device.
- Flora [20]. The Flora board is a small (1.75" diameter, 4.4 grams) wearable electronic platform. It is a typical representation of many Arduino compatible devices. This board only contains a micro-controller and programming circuits, without using any batteries and sensors. Flora is also a possible candidate for our purposes.
- GENEActiv Wireless [21]. This company offers four variations of bracelets (wristband) for humans. Depending on the variant, it includes an accelerometer, body temperature sensor, skin temperature sensor and a gyroscope, only the wireless version is equipped with live wireless data transmission via a Bluetooth interface. The built-in three-axis gyroscope allows readings up to 1 kHz. The battery life ranges from 8 hours (wireless version) to 45 days (original version).
- MbientLab MetaWear [9]. This is a smart sensor module for motion tracking and contains an accelerometer, gyroscope, pressure, thermometer sensor, and special mathematical preprocessor to combine data from each sensor integrated on the board. This device can be paired with an BLUE gateway hub to transmit data to the cloud.
- Fitbit [22] is another human activities' tracker. It is small (e.g. wrist or pocket size) and contains various sensors to track steps and estimate energy consumption, sleep and movement actions. The battery life is around 3 days and the device can store accumulated data of the time span of 1 week. The Fitbit device transfers data wireless to a base station which is also a charging device. This device is a great example of popular wearable devices in the market.
- Xiaomi MiBand [23]. This is the fitness tracker for humans. It contains an accelerometer and notification LEDs. With the original manufacturer's firmware, this device only provides aggregated data. The battery can last up to 30 days.

B. Proposed project

Although smart pet systems with the targeted specifications and objectives start to appear on the market, they are often mono-function stand-alone devices such as a single health monitor, activities tracker or feed dispenser. They are also mostly heavy and require strong power supply sources such as big battery packs or power lines. Existing systems are not designed to connect and control other smart toys for pets. In addition, they cannot be integrated into existing smart voice assistant system such as Google Home or Apple Siri. Hence, users still end up having multiple systems to handle, which is undesirably hindering from gain interest for such systems among pet-owners.

The proposed study aims to contribute to overcome these issues and offer multi-use smart products for pets (owners).

In this study, we propose an ecosystem for pets' toys consisting of a collar, toys, and a mobile application to communicate with the collar.

The proposed collar is designed to contain/sense ambient temperature, magnet field, acceleration, BLE and a speaker.

Currently, the collar prototype does not have as many sensors and functionalities as the aforementioned devices, but we will gradually improve its design and application.

Our proposed software is fed by collected data from a collar for monitoring purposes. These data are motion, ambient temperature, and magnetic proximity. The motion data are gathered from the acelerometer sensor, while proximity data between the collar and designated objects are measured via a magnetic field sensor. The environment temperature is transmitted by an ambient sensor. All those sensors transmit data via BLE to the receiver (e.g., mobile phone application). This information could also be used to encourage specific behavior of pets for training purposes. For example, to prevent the pet, (e.g. a dog), to jump inside the living room, the owner can activate an external device (e.g. a food dispenser machine to provide foods) whenever the system detects that the dog is jumping, which distracts the pet. To fulfill these objectives, it is needed to develop a system that can keep track and record data of pets (e.g. movements,).

Based on collaboration with BeOneBreed - a local industrial partner - who possesses extensive expertise in making toys for pet, we are developing a prototype system to interact with small pets such as cats or dogs. The proposed system includes a collar attached to a pet, which interacts with other peripheral devices such as a food dispenser, interactive moving toys, etc. An early stage prototype of the collar was provided by BeOneBreed.

C. Project's objectives

The project aims to create an ecosystem of connectivity products (IoT) for pets. This ecosystem can include a vast range of product types from feeding, tracking, health monitoring to interactive toys for animals.

A key aspect of this main objective is to create a system of smart devices for pets, which integrates with existing smart home systems. It will provide the software layer for communicating and exchanging data as well as sending control commands between many smart home objects and systems such as Google Nest or Amazon Alexa. This software layer acts like a proxy to translate high level commands to BLE commands to access raw data on the collar. It also convert raw sensor data to common formats such as CSV or JSON. For example, the pet's owner asks Alexa: "what is my dog doing?". Alexa translate that sentence to a specific command set and sends to the BLE hub (or mobile application) which acts as the middle sofware layer. The BLE hub translates again that high level commands to BLE commands to get the collar data to provide the predictions. Then it provides the high level answer back to Alexa which in turn speaks to the owner.

We aim to develop an integrated ecosystem of pets' smart products including a collar and a wide range of connectivity devices (e.g., toys) focusing on the following key features:

- The wearable items are robust and lightweight.
- The wearable items have low power consumption.
- The wearable device must be able to connect to multiple devices (e. g., 'smart toys') in real-time.
- The communication between the wearable and the non-wearable devices must be fast, and reliable.
- The time windows of data exchange between devices must be minimized because of the non-stationary nature of the pets. The size of data also must be minimized (e.g., compressed).
- These products can communicate with each other and with current smart home systems. Such systems often include a collar attached to the pets, and several connected entities such as a food dispenser, and other toys.

At the same time, the project also aims to improve human-animal interaction by understanding their pets' behaviors via connectivity products. The data are collected from a portable device attached the pets and other connected devices installed inside a domestic environment. Based on those data, the system allows pet owners to decide best actions to take according to the pet's behaviour. In order to do that, the proposed mobile application collects and displays pet motion behaviors to inform owners about abnormal motion patterns of their pets. The details of this step are described in Section III-D of the System Design and Discussion part (III).

Our proposed system will include:

- A range of communicative products allowing better understanding of the animal's behavior, to assist identification of their habits enabling a medical and/or behavioral diagnosis. Thus, the system detects and records the events and activities of the animal.
- A range of communicative products assisting to monitor pets' behavior to optimize care by creating interactions. The system has to be interactive and trigger remotely programmed events.
- A range of communicative products which are easy to use, either for installation or handling (application software must be user-friendly).

In addition to the main objectives, specific interactions between the products are essential to the success of the project:

- Direct interaction between collar and accessories;
- Possibility to have multiple collars in the same area and identification of each of them;
- Direct interaction between accessories and the user (via an app) provided by wireless technology;
- No direct interaction between accessories and internet;
- Interaction with internet only from the user (via an app);
- User-specific configurable functions (via an app);
- Ability to add an interaction between the accessories.

On the other hand, each of the communicative products also has specific features. The function can allows us to propose a modular technical approach, with the development of three electronic modules, either an electronic module specific to the collar, a generic module for the ephemeral objects, and a module specific for each of the peripheral objects.

II. Methodology

We define the following principles for system development based on the projects' objectives:

- Optimize the software for wearable devices: based on the hard- and software specifications of the device, create 1) the core software interface (i.e. APIs) to communicate with the device via Bluetooth from the mobile application, 2) develop the control commands for the mobile application with functionalities such as reading, writing, storing data periodically from the connected collar to the application, then 3) create the reporting functions based on collected data.
- Minimize energy and resources consumption at a local device by moving all data to the cloud for heavy processing: including 1) design the software architecture and technologies to coordinate the cloud computing and the mobile application, 2) develop the ability to sync (e.g. downloading and uploading) data to the cloud from the mobile application 3) develop the interface for data management functionalities on the cloud server.
- Maximize the integration and expansion abilities of the system: based on the requirements, 1) propose a possible

IoT gateway to communicate with connected collars, 2) propose the functionalities of the gateway; the gateway must be able to connect with multiple collars and other future 'smart toys' of BeOneBreed, and 3) propose the solution to connect the IoT gateway to an existing smart home controller service like Google Home to command the device.

III. System design and discussion

The main goal is to create an ecosystem for smart toys. It has the collar which plays a central role for communication and data collections as well as dashboard display. Furthermore, it is possible to have a BLE hub to WiFi gateway which can be used when there is no mobile application available. The hub has the similar functions as the mobile application. Figure 1 shows an overview of the entire ecosystem.

A. Proposed system architecture

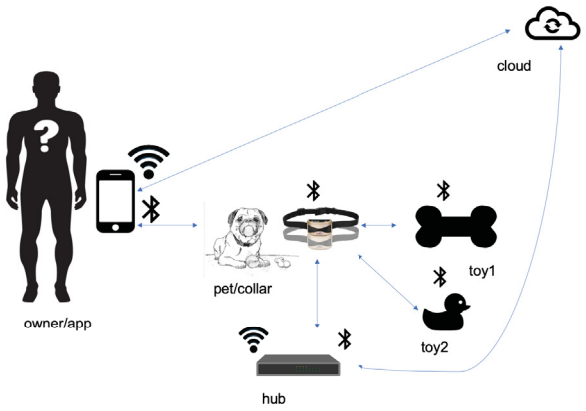

Fig. 1. Overview of the ecosystem and its components.

Based on the abstraction level of the ecosystem, we proposed an architecture, including both software and hardware components for the project, as outlined in Figure 2.

The overview of each component and its functionalities is detailed below.

- The toy: is a Bluetooth connected device, which communicates directly with the collar and the hub. It sends the data and can be triggered by the collar.
- The mobile application: is the main user control application. The pet owner uses it to see information about their pet's activities. The main functions of the application are to trigger commands in the collar and send/retrieve data from the cloud server.
- The cloud: is the main data storage and data processing unit. After having historical data from the pet's behaviors, the analyzing procedures are executed in the cloud to perform activity prediction and other data processing. The results are transferred to the mobile application for presentation to end users.

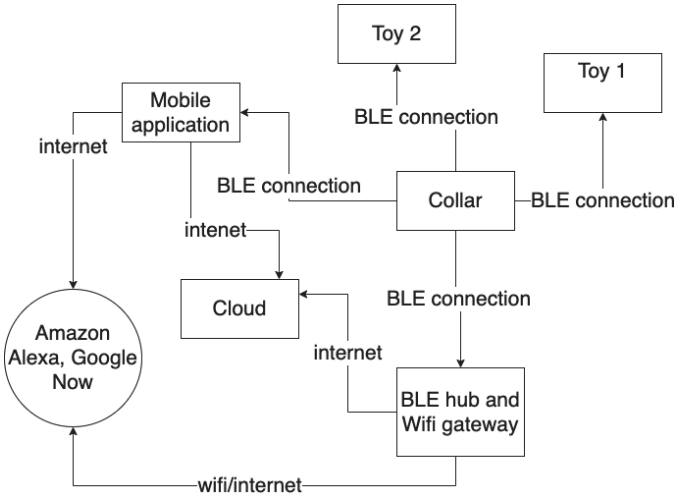

Fig. 2. Abstracted view of the system architecture. The arrows indicate the data connection between components.

- BLE hub and WiFi gateway: when the users do not have the mobile application to collect data, this gateway will handle this part and send all those data to the cloud. Moreover, it is the connection point for smart voice assistant systems to control the entire smart toy ecosystem.

B. Early collar prototype

The collar is the centre of the system. BeOneBreed provided a concept prototype that can be used. This preliminary collar prototype uses the nRF52832 ARM Cortex-M4F in form of the BC832 miniature module from Fanstel 3. The outer case components are produced by a 3-D printer. The collar features a wireless charging battery pack, its weight is about 50 gram with standard configuration and it has 3 types of sensors: an accelerometer sensor, a magnet field sensor, and an ambient temperature sensor. The collar communicates via BLE 5.0 and has a firmware developed by Motsai[1].

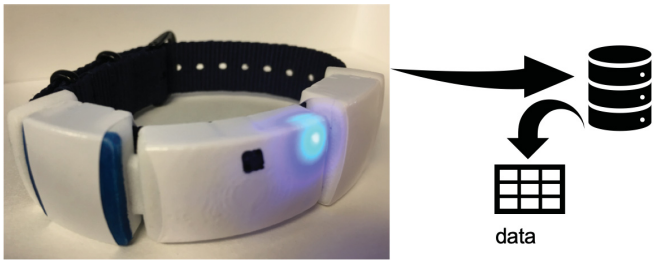

Fig. 3. The preliminary prototype collar as designed by Motsai. Currently, only raw sensor data is stored.

C. Proposed mobile application

As a starting point, we develop a mobile application connecting with the collar to provide the most crucial functions:

[1]http://motsai.ca

- The application collects the data from the sensors in real-time when in range.
- The application provides the graphs and aggregated statistic information from the data.
- The application will be able to run on iOS or Android based phones.
- The application can store and synchronize data with the cloud database.
- Extended functions: can control the collar via digital voice assistance.

Figure 4 outlines a sketch of the mobile application with its core functions and hardware and data requirements. It features three key components: the collar, the cloud and the mobile application.

The first component is the collar. It collects the data from the sensors. For example, the accelerometer data show the moving speed of the pets and the temperature sensor retrieves the relatively skin thermal level of the pets. These data can be sent to the application in real-time or in small quantity.

The second component is the cloud. This is the place where long-term data are stored and computation of intensive tasks is executed. For example, prediction of abnormal patterns of pets based on monthly activities using deep learning.

The third part is the mobile application. Its main goals are to interact with the user. It acquires data directly from the collar via Bluetooth protocols and it receives data from the cloud for analysis. The application sends its data to the cloud.

To realize the mobile application functionalities, it is needed to feed the data from the other components. For example, to be able to display pet's behaviour statistics, the mobile software parts need to access the historical data from the cloud. Those data are collected by the collar from the built-in sensors. The processing unit at the cloud is key: it uses algorithms to predict what activities the pet has performed. The mobile application displays the analytic results in an expressive way to pet's owner.

All the basic functions such as download data from the collar, start and stop the data recording are implemented.

Auto-activity classification based on motion data will be added in a next step: the goal is to display a summary of daily or hourly activities of the pet by using accelerometer data.

D. Pets' behavior classification on mobile device and collar

- Data preparation: in this step, we collaborate with the industrial partner to collect the data. The partner has a controlled environment with a video camera and the pet inside. The camera records the motion activities of the pet while the collar collects measurements from the accelerometer sensor. By observing the pet actions, the partner creates a labelled dataset by associating the pet motion with the sensor data at the same time point. Later, we use this data for modeling in further steps.
- Behavior classification on a mobile application: we are using the aggregated data on days and hours to the classification. Despite to the limitation of training and labeling data, the supervised machine learning method is used to classify the motion intensity and assign them as: stationary (e.g. sleeping, resting), moderate movement (e.g. normal walking inside the house) and fast moving (e.g. running). This procedure is performed automatically without user intervention. In particularly, we are experimenting standard machine learning algorithm such as linear regression [24], random forest classifier [25], suport-vectors machine [26] and deep neural network (e.g, LSTM) [27]. For current data, it shows that random forest performs better than the others.
- Behavior classification on the collar: the previous approach has some limitations such as requiring relatively big storage memory for raw sensor's data, running completely offline. To overcome such issues, we propose another embedded application to classify motion activities of the pet. This embedded application runs in real-time and produces instantly prediction about the pet's behavior at a specific time . Now, only the prediction data are stored inside the collar, which significantly reduce the memory demand. However, the classification application requires a trained model. This trained model must be produced by a supervised learning algorithm running on an external device such as computer. It also requires enough training and labeling data to be able to predict accurately.

IV. Conclusion and results

A. Current results

We have proposed a smart toy's ecosystem for pets which is robust to change, extensible and well-integrated in existing smart home systems. It is possible to add new toys to the system, as long as the new toys are BLE compatible. We also designed a system architecture including software architecture and hardware components. With the help of the cloud, the smart collar can be controlled over general Internet via the mobile application and the hub. This can be useful for remote monitoring and detection of abnormal activities. The preliminary collar prototype is a lightweight wearable IoT device for pets and will have the ability to be programmed for complex processing in next steps. In addition, it has a mobile application capable of recording sensor data from the collar and controlling its basic functions. For better understanding our work, we provide an brief summary of steps to achieve the project goals:

1) Collect hardware requirements (done by industrial partner)
2) Collect software requirements
3) Hardware implementation (out-source)
4) Software design: system architecture design, mobile application design, BLE hub design
5) Software implementation: mobile application, cloud storage implementation, BLE hub

When implementing this project, we face some obstacles:

1) Have to iterate many round to get the suitable requirements of the hardware and software.

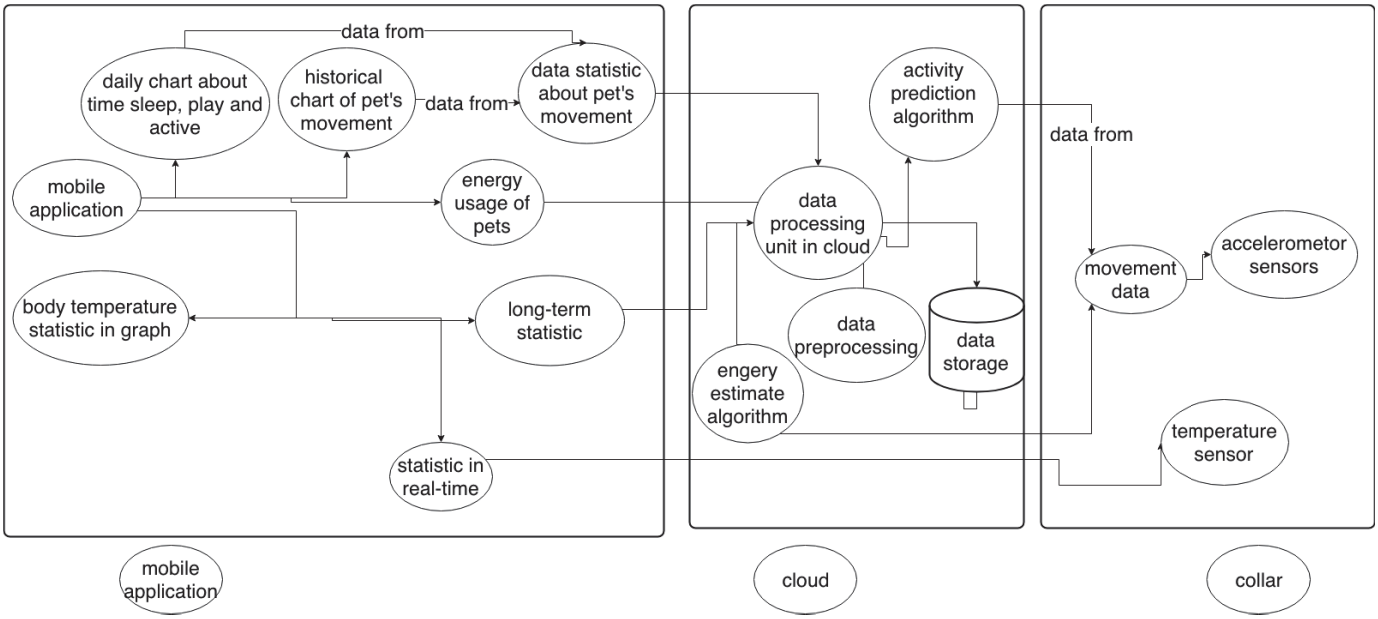

Fig. 4. The mutual connection between mobile application and other components.

2) Whenever requirements change, we have to adapt the software to its purpose. This process is not optimal but enough for a working prototype
3) The battery time is short relatively to the requirements. It can be increased by energy-optimization-ed firmware and equipping more battery cells.

B. Future works

In the next phase of the project, extended functionality of the mobile application will be developed, e.g. unexpected behavior pattern detection, based on the sensor data synchronized to the cloud. Using long-term history data of a pet and machine learning approaches, insights will be extracted to better understand the well-being of pets.

Furthermore, to increase the efficiency of data storage and battery life, a short-term behavior classification application will be developed and deployed in the collar. This embedded software will run in a real-time manner on the current sensor data to store pet motions prediction in the collar. Then it is not required to store raw sensor data, allowing the internal memory (8 MB) to last for two years. Each minute one predicated motion will be stored, consuming 20 bytes[2]. This results in an extended battery life from hours to weeks improving significantly the user experience of the device in the smart ecosystem.

ACKNOWLEDGMENT

The authors would like to thank the team of BeOneBreed[3] for the fruitful discussions, their support and initiation of this project. This research is funded by a Mitacs Accelerate program. We also thank Motsai[4] and Dr. Rolf Wuthrich for the fruitful technical discussion which helped to accelerate the project.

REFERENCES

[1] Juliet Clutton-Brock. *A natural history of domesticated mammals*. Cambridge University Press, 1999.
[2] James Serpell and Priscilla Barrett. *The domestic dog*. Cambridge University Press, 2016.
[3] Jessica Gall Myrick. Emotion regulation, procrastination, and watching cat videos online: Who watches internet cats, why, and to what effect? *Computers in human behavior*, 52:168–176, 2015.
[4] Yuval Noah Harari. A brief history of humankind. *Publish in agreement with The Deborah Harris Agency and the Grayhawk Agency*, 2014.
[5] American pet products association, inc. https://www.americanpetproducts.org. (Accessed on 10/21/2019).
[6] Facts + statistics: Pet statistics.
[7] Pamela N. Danziger. Pets Are Going Digital: The Brands Pioneering The $565 Million Market For Smart Pet Products.
[8] Pet ownership and human health: a brief review of evidence and issues.
[9] McNicholas et al. Pet ownership and human health: a brief review of evidence and issues. *Bmj*, 331(7527):1252–1254, 2005.
[10] Mookkie — the pet bowl with a.i. https://www.mookkie.com/. (Accessed on 10/17/2019).
[11] Sureflap — award winning microchip cat doors & feeders. https://www.surepetcare.com/en-US. (Accessed on 10/17/2019).

[2]20 bytes is the minimum size of each sample
[3]https://www.beonebreed.com
[4]http://motsai.com

[12] Smart dog collar - heart rate monitor & more — petpace. https://petpace.com/. (Accessed on 10/17/2019).
[13] Fitbark 2 dog activity monitor — fitbark. https://www.fitbark.com/store/fitbark2/. (Accessed on 10/17/2019).
[14] Scollar the smart pet collar — scollar makes caring for your pets as easy as loving them. https://www.scollar.com/. (Accessed on 10/17/2019).
[15] Juneyoung Ahn et al. Pet buddy: A wearable device for canine behavior recognition using a single imu. In *2016 International Conference on Big Data and Smart Computing (BigComp)*, pages 419–422. IEEE, 2016.
[16] Gary Weiss et al. Wagtag: a dog collar accessory for monitoring canine activity levels. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*, pages 405–414. ACM, 2013.
[17] David Sec et al. System for detailed monitoring of dog's vital functions. In *International Conference on Computational Collective Intelligence*, pages 426–435. Springer, 2018.
[18] Ying Guo et al. Animal behaviour understanding using wireless sensor networks. In *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*, pages 607–614. IEEE, 2006.
[19] oblu iot – oblu - an open platform for wearable motion sensing. https://www.oblu.io/. (Accessed on 10/17/2019).
[20] Flora - wearable electronic platform: Arduino-compatible [v3] id: 659 - $14.95 : Adafruit industries, unique & fun diy electronics and kits. https://www.adafruit.com/product/659. (Accessed on 10/17/2019).
[21] Wrist-worn accelerometer research watches — activinsights. https://www.activinsights.com/products/geneactiv/. (Accessed on 10/17/2019).
[22] Fitbit official site for activity trackers & more. https://www.fitbit.com/home. (Accessed on 10/21/2019).
[23] Mi global home. https://www.mi.com/global/miband. (Accessed on 10/17/2019).
[24] Sanford Weisberg. *Applied linear regression*, volume 528. John Wiley & Sons, 2005.
[25] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
[26] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.
[27] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

APPENDIX II

MOBILE APPLICATION SOURCE CODE

The full source code of the mobile application is hosted at https://github.com/vantienvincent/Tesis.

The code listing II.1 is an extraction from the **SensorRawDataReader.swift** file that was used to read the raw sensor data.

```
1  func readSensorBinaryData(_ filePath: String) -> (NSDate, [(Double,Int16,
       Int16, Int16)],
2       [(Int16, Int16, Int16)] ){
3       print("file is: \(filePath)")
4       var accWithTimestamp : [(Double,Int16, Int16, Int16)] = []
5           var accNoTimeStamp : [(Int16, Int16, Int16)] = []
6       let data = FileManager.default.contents(atPath: filePath)
7       var x = data?.subdata(in: 0..<496) // this is the header
8       // at this position, y is the time the sensor data is recorded
9       var timeStamp = Double(x![5-1]) + Double(x![6-1])*256 + Double(x![7-1])
       *pow(2,16) + Double(x![8-1])*16777216;
10      let date = NSDate(timeIntervalSince1970: timeStamp)
11      var index = 496
12      // index starting from the number 496
13      while index < data!.count{
14          // timestamp 3 bytes + 2 bytes ax + 2 bytes ay + 2 bytes az = 9
       bytes, 12 is for safety
15          if index + 12 > data!.count{
16              break
17          }
18          // read 3 control bytes
19          x = data?.subdata(in: index..<index+3)
20          let m = x! as NSData
21
22          //convert bytes to data
23          let k:Data = Data(bytes: m.bytes, count: 3)
24          //print(x)
```

```
25          let number: Int32 = k.withUnsafeBytes {
26              (pointer: UnsafePointer<Int32>) -> Int32 in
27              return pointer.pointee // reading four bytes of data
28          }
29          var bytes = [UInt8](x!)
30          index = index + 3
31          var acc_nb_packets = 0
32          if bytes[0] == 13 && bytes[2] == 16 || bytes[0]==1 && bytes[2] ==
    31
33          {
34          }
35          else if bytes[0]==13 && bytes[2] == 11{
36          }
37          else if bytes[0]==1 && bytes[2] == 4{
38          }
39          else if bytes[0]==1 && bytes[2] == 6 {
40          }
41          else if bytes[0]==13 && bytes[2] == 13 {
42          }
43          else if bytes[0]==1 && bytes[2] == 5 {
44          }
45          else if bytes[0]==13 && bytes[2] == 15 {
46          }
47          else if bytes[0] == 13 && bytes[2] == 10{ // this is accelerometer
    data
48              acc_nb_packets = acc_nb_packets + 1
49              // get next 4 bytes for time stamp
50              var x = data?.subdata(in: index..<index + 4)//timestamp
51              var k:Data = Data(bytes: m.bytes, count: 4)
52              timeStamp = Double(x![0]) + Double(x![1])*256 + Double(x![2])*
    pow(2,16) + Double(x![3])*16777216;
53              index = index + 4
54              x = data?.subdata(in: index..<index + 2)//ax
55              var m = x! as NSData
56              k = Data(bytes: m.bytes, count: 2)
```

```
57              let ax: Int16 = k.withUnsafeBytes {
58                  (pointer: UnsafePointer<Int16>) -> Int16 in
59                  return pointer.pointee // reading four bytes of data
60              }
61              //ay
62              index = index + 2
63              x = data?.subdata(in: index..<index + 2)//ay
64              m = x! as NSData
65              k = Data(bytes: m.bytes, count: 2)
66              let ay = k.withUnsafeBytes {
67                  (pointer: UnsafePointer<Int16>) -> Int16 in
68                  return pointer.pointee // reading four bytes of data
69              }
70              //az
71              index = index + 2
72              x = data?.subdata(in: index..<index + 2)//az
73              m = x! as NSData
74              k = Data(bytes: m.bytes, count: 2)
75              let az = k.withUnsafeBytes {
76                  (pointer: UnsafePointer<Int16>) -> Int16 in
77                  return pointer.pointee // reading four bytes of data
78              }
79              index = index + 2
80              accWithTimestamp.append((timeStamp, ax, ay, az))
81              accNoTimeStamp.append((ax, ay, az))
82          }
83      }
84      return (date, accWithTimestamp, accNoTimeStamp)
85  }
```

Listing II.1: Code snippet for reading raw sensor data

APPENDIX III

RAW SENSOR DATA STRUCTURE

The accelerometer sensor data are stored in a binary format according to the design of the hardware vendor. The simple structure of this binary data file is as following:

1. File header: 496 bytes
2. Timestamp: 4 bytes - time the file is created
3. For each data sample
    a. Timestamp: 3 bytes - time the sample is recorded
    b. Control bytes: 3 bytes - for reserve purpose
    c. x value: 2 bytes
    d. y value: 2 byes
    e. z value: 2 bytes

    x,y,and z values indicate the motion levels in a 3D space.

LIST OF BIBLIOGRAPHICAL REFERENCES

Activinsights. (2019, June, 2). Wrist-Worn Accelerometer Research Watches | Activinsights. Retrieved from https://www.activinsights.com/products/geneactiv/.

Ahn, J. et al. (2016). Pet Buddy: A wearable device for canine behavior recognition using a single IMU. *2016 International Conference on Big Data and Smart Computing (BigComp)*, pp. 419–422.

Anderson, E. (1936). The species problem in Iris. *Annals of the Missouri Botanical Garden*, 23(3), 457–509.

Anguita, D., Ghio, A., Oneto, L., Parra, X. & Reyes-Ortiz, J. L. (2012). Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. *International workshop on ambient assisted living*, pp. 216–223.

Anguita, D., Ghio, A., Oneto, L., Parra, X. & Reyes-Ortiz, J. L. (2013). A public domain dataset for human activity recognition using smartphones. *Esann*, 3, 3.

Apple, I. (2020, August, 2). Integrate machine learning models into your app. [html]. Retrieved from https://developer.apple.com/documentation/coreml.

Arora, M. & Kansal, V. (2019). Character level embedding with deep convolutional neural network for text normalization of unstructured data for Twitter sentiment analysis. *Social Network Analysis and Mining*, 9(1), 12.

Association, A. P. P. (2019, June, 2). American Pet Products Association, Inc. Retrieved from https://www.americanpetproducts.org.

Barber, D. (2012). *Bayesian reasoning and machine learning*. Cambridge University Press.

Barnes, R. et al. [Cats Domesticated Themselves]. (2018, Apr, 2). Cats Domesticated Themselves, Ancient DNA Shows. Retrieved from https://www.nationalgeographic.com/news/2017/06/domesticated-cats-dna-genetics-pets-science/.

Bayat, A., Pomplun, M. & Tran, D. A. (2014). A study on human activity recognition using accelerometer data from smartphones. *Procedia Computer Science*, 34, 450–457.

Ben-Hur, A., Horn, D., Siegelmann, H. T. & Vapnik, V. (2000). A support vector clustering method. *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, 2, 724–727.

Ben-Hur, A., Horn, D., Siegelmann, H. T. & Vapnik, V. (2001). Support vector clustering. *Journal of machine learning research*, 2(Dec), 125–137.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Borodin, A., Mirvoda, S., Kulikov, I. & Porshnev, S. (2017). Optimization of memory operations in generalized search trees of PostgreSQL. *International Conference: Beyond Databases, Architectures and Structures*, pp. 224–232.

Bujari, A., Licar, B. & Palazzi, C. E. (2012). Movement pattern recognition through smartphone's accelerometer. *2012 IEEE Consumer Communications and Networking Conference (CCNC)*, pp. 502–506.

Carroll, A., Heiser, G. et al. (2010). An analysis of power consumption in a smartphone. *USENIX annual technical conference*, 14, 21–21.

Caruana, R. & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. *Proceedings of the 23rd international conference on Machine learning*, pp. 161–168.

CENTER, P. R. (2020, August, 20). Smartphone Ownership Is Growing Rapidly Around the World, but Not Always Equally [html]. Retrieved from https://www.pewresearch.org/global/2019/02/05/smartphone-ownership-is-growing-rapidly-around-the-world-but-not-always-equally.

Chaudhuri, T. et al. (2018). Random forest based thermal comfort prediction from gender-specific physiological parameters using wearable sensing technology. *Energy and Buildings*, 166, 391–406.

Chawla, N. V. (2009). Data mining for imbalanced datasets: An overview. In *Data mining and knowledge discovery handbook* (pp. 875–886). Springer.

Chen, D., Yuan, Z., Hua, G., Zheng, N. & Wang, J. (2015). Similarity learning on an explicit polynomial kernel feature map for person re-identification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1565–1573.

Chen, T.-S., Li, X.-G., Hsieh, K.-H., Xiao, Y. & Shang-Hui, P. (2014). Smart gateway, smart home system and smart controlling method thereof. 2. US Patent App. 14/071,569.

Chinchor, N. (1992). MUC-4 Evaluation Metrics. *Proceedings of the 4th Conference on Message Understanding*, (MUC4 '92), 22–29.

Clevenger, K. A., Pfeiffer, K. A., Mackintosh, K. A., McNarry, M. A., Brønd, J., Arvidsson, D. & Montoye, A. H. (2019). Effect of sampling rate on acceleration and counts of hip-and wrist-worn ActiGraph accelerometers in children. *Physiological measurement*, 40(9), 095008.

Clutton-Brock, J. (1999). *A natural history of domesticated mammals*. Cambridge University Press.

Connect, F. (2020, March, 2). SureFlap | Award Winning Microchip Cat Doors & Feeders. Retrieved from https://www.surepetcare.com/en-US.

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273–297.

Derguech, W., Bruke, E. & Curry, E. (2014). An autonomic approach to real-time predictive analytics using open data and internet of things. *2014 IEEE 11th Intl Conf on Ubiquitous Intelligence and Computing and 2014 IEEE 11th Intl Conf on Autonomic and Trusted Computing and 2014 IEEE 14th Intl Conf on Scalable Computing and Communications and Its Associated Workshops*, pp. 204–211.

Do, C.-T., Douzal-Chouakria, A., Marié, S. & Rombaut, M. (2015). Multiple Metric Learning for large margin kNN Classification of time series. *2015 23rd European Signal Processing Conference (EUSIPCO)*, pp. 2346–2350.

Electronics. (2020, December, 2). Accelerometer boasts ultra-low power consumption [html]. Retrieved from https://www.electronicspecifier.com/products/sensors/accelerometer-boasts-ultra-low-power-consumption.

FitBark. (2019, June, 2). FitBark 2 Dog Activity Monitor | FitBark. Retrieved from https://www.fitbark.com/store/fitbark2/.

Fitbit. (2019, June, 2). Fitbit Official Site for Activity Trackers & More. Retrieved from https://www.fitbit.com/home.

FLORA. (2019, June, 2). FLORA - Wearable electronic platform: Arduino-compatible [v3] ID: 659 - $14.95 : Adafruit Industries, Unique & fun DIY electronics and kits. Retrieved from https://www.adafruit.com/product/659.

Foerster, F. & Fahrenberg, J. (2000). Motion pattern and posture: correctly assessed by calibrated accelerometers. *Behavior research methods, instruments, & computers*, 32(3), 450–457.

Fortune, E., Lugade, V., Morrow, M. & Kaufman, K. (2014). Validity of using tri-axial accelerometers to measure human movement–Part II: Step counts at a wide range of gait velocities. *Medical engineering & physics*, 36(6), 659–669.

Frantz, L. A. et al. (2016). Genomic and archaeological evidence suggest a dual origin of domestic dogs. *Science*, 352(6290), 1228–1231.

Friedman, J., Hastie, T. & Tibshirani, R. (2001). *The elements of statistical learning*. Springer series in statistics New York.

García-Martín, E., Rodrigues, C. F., Riley, G. & Grahn, H. (2019). Estimation of energy consumption in machine learning. *Journal of Parallel and Distributed Computing*, 134, 75–88.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep learning*. MIT press.

Guo, Y. et al. (2006). Animal behaviour understanding using wireless sensor networks. *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*, pp. 607–614.

Haladjian, J., Ermis, A., Hodaie, Z. & Brügge, B. (2017). iPig: Towards tracking the behavior of free-roaming pigs. *Proceedings of the Fourth International Conference on Animal-Computer Interaction*, pp. 1–5.

Haladjian, J., Haug, J., Nüske, S. & Bruegge, B. (2018). A wearable sensor system for lameness detection in dairy cattle. *Multimodal Technologies and Interaction*, 2(2), 27.

Harari, Y. N. (2014). *Sapiens. A Brief History of Humankind*. London: Vintage Books.

He, H. & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9), 1263–1284.

Hilbe, J. M. (2009). *Logistic regression models*. CRC press.

Hof, L. & Hoang, V. T. (2020, June). *Towards a Smart Toy Ecosystem for Pets*. Mechanica presented in Canadian Society for Mechanical Engineering International Congress 2020, Charlottetown PE, Canada.

Hu, F., Silver, D. & Trude, A. (2008). LonelyDog@Home. 333–337. doi: 10.1109/wiiatw.2007.74.

Hu, S. (2015). Research on data fusion of the internet of things. *2015 International Conference on Logistics, Informatics and Service Sciences (LISS)*, pp. 1–5.

Institute, I. I. (2019, June, 2). Facts + Statistics: Pet statistics. Retrieved from https://www.iii.org/fact-statistic/facts-statistics-pet-statistics.

Jagadish, H. V., Ooi, B. C., Tan, K.-L., Yu, C. & Zhang, R. (2005). iDistance: An adaptive B+-tree based indexing method for nearest neighbor search. *ACM Transactions on Database Systems (TODS)*, 30(2), 364–397.

Jalal, A., Quaid, M. A. K. & Hasan, A. S. (2018). Wearable sensor-based human behavior understanding and recognition in daily life for smart environments. *2018 International Conference on Frontiers of Information Technology (FIT)*, pp. 105–110.

Kaelbling, L. P., Littman, M. L. & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4, 237–285.

Karantonis, D. M., Narayanan, M. R., Mathie, M., Lovell, N. H. & Celler, B. G. (2006). Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring. *IEEE transactions on information technology in biomedicine*, 10(1), 156–167.

Karatzoglou, A., Smola, A., Hornik, K. & Zeileis, A. (2004). kernlab-an S4 package for kernel methods in R. *Journal of statistical software*, 11(9), 1–20.

Kumpulainen, P. et al. (2018). Dog activity classification with movement sensor placed on the collar. *ACM International Conference Proceeding Series*, 3–8.

Ladha, C. et al. (2013a). Dog's life: Wearable activity recognition for dogs. *UbiComp 2013 - Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, (September), 415–418. doi: 10.1145/2493432.2493519.

Ladha, C. et al. (2013b). Dog's life: wearable activity recognition for dogs. *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 415–418.

Lane, N. D., Bhattacharya, S., Georgiev, P., Forlivesi, C. & Kawsar, F. (2015). An Early Resource Characterization of Deep Learning on Wearables, Smartphones and Internet-of-Things Devices. *Proceedings of the 2015 International Workshop on Internet of Things towards Applications*, (IoT-App '15), 7–12.

LavvieBot. (2020, June, 2). LavvieBot S. Retrieved from https://www.lavviebot.com/en/home.html.

Lee, D. S., Chong, T. W. & Lee, B. G. (2016). Stress events detection of driver by wearable glove system. *IEEE Sensors Journal*, 17(1), 194–204.

Lester, J., Choudhury, T. & Borriello, G. (2006). A practical approach to recognizing physical activities. *International conference on pervasive computing*, pp. 1–16.

Lindemann, A., Schnor, B., Sohre, J. & Vogel, P. (2016). Indoor Positioning: A Comparison of WiFi and Bluetooth Low Energy for Region Monitoring. *HEALTHINF*, pp. 314–321.

Liu, X., Xuan, J., Hussain, F. & Chong, C. (2019). ARM-based Behavior Tracking and Identification System for Grouphoused Pigs. *Recent Advances in Electrical & Electronic Engineering (Formerly Recent Patents on Electrical & Electronic Engineering)*, 12(6), 554–565.

Lugade, V., Fortune, E., Morrow, M. & Kaufman, K. (2014). Validity of using tri-axial accelerometers to measure human movement—Part I: Posture and movement detection. *Medical engineering & physics*, 36(2), 169–176.

Ma, Z., Qiao, Y., Lee, B. & Fallon, E. (2013). Experimental evaluation of mobile phone sensors.

Mahadevan, S. & Connell, J. (1992). Automatic programming of behavior-based robots using reinforcement learning. *Artificial intelligence*, 55(2-3), 311–365.

Mannini, A. & Sabatini, A. M. (2010). Machine learning methods for classifying human physical activity from on-body accelerometers. *Sensors*, 10(2), 1154–1175.

mbientlab. (2019, June, 2). MbientLab Wearable Bluetooth 9-axis IMUs environmental Sensors. Retrieved from https://mbientlab.com.

McNicholas et al. (2005). Pet ownership and human health: a brief review of evidence and issues. *Bmj*, 331(7527), 1252–1254.

Mi. (2019, June, 2). Mi Global Home. Retrieved from https://www.mi.com/global/miband.

Microsoft. (2019, June, 2). Microsoft. Retrieved from https://support.microsoft.com/en-us/help/4467073/end-of-support-for-the-microsoft-health-dashboard-applications.

Mookkie. (2019, November, 17). Mookkie | The Pet Bowl with A.I. [html].

Myrick, J. G. (2015). Emotion regulation, procrastination, and watching cat videos online: Who watches Internet cats, why, and to what effect? *Computers in human behavior*, 52, 168–176.

NCBI. (2019, June, 2). Pet ownership and human health: a brief review of evidence and issues. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1289326/.

oblu IoT. (2019, June, 2). oblu IoT – oblu - an open platform for wearable motion sensing. Retrieved from https://www.oblu.io/.

Pal, M. (2005). Random forest classifier for remote sensing classification. *International journal of remote sensing*, 26(1), 217–222.

Papagiannaki, A. et al. (2019). Recognizing physical activity of older people from wearable sensors and inconsistent data. *Sensors (Switzerland)*, 19(4), 1–19.

Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, 12, 2825–2830.

Petpace. (2019, June, 2). Smart Dog Collar - Heart Rate Monitor & More | Petpace. Retrieved from https://petpace.com/.

Pongrácz Rossi et al. (2016). A dog using skype. *ACM International Conference Proceeding Series*, 15-17-Nove, 1–4.

Putra, G. D., Pratama, A. R., Lazovik, A. & Aiello, M. (2017). Comparison of energy consumption in Wi-Fi and bluetooth communication in a Smart Building. *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 1–6.

Rice, S. (1950). Communication in the presence of noise—Probability of error for two encoding schemes. *Bell System Technical Journal*, 29(1), 60–93.

Robert, C. (2014). *Machine learning, a probabilistic perspective*. Taylor & Francis.

Ronneberger, O., Fischer, P. & Brox, T. (2015a). U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241.

Ronneberger, O., Fischer, P. & Brox, T. (2015b). U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241.

Sanders, R. (1987). The Pareto principle: its use and abuse. *Journal of Services Marketing*.

Scollar. (2019, June, 2). Scollar The Smart Pet Collar | Scollar makes caring for your pets as easy as loving them. Retrieved from https://www.scollar.com/.

Seber, G. A. & Lee, A. J. (2012). *Linear regression analysis*. John Wiley & Sons.

Sec, D. et al. (2018a). *System for Detailed Monitoring of Dog ' s Vital Functions*. Springer International Publishing.

Sec, D. et al. (2018b). System for Detailed Monitoring of Dog's Vital Functions. *International Conference on Computational Collective Intelligence*, pp. 426–435.

Serpell, J. & Barrett, P. (2016). *The domestic dog*. Cambridge University Press.

Shannon, C. E. (1998). Communication in the presence of noise. *Proceedings of the IEEE*, 86(2), 447–457.

Shen, L., Lou, Y., Chen, Y., Lu, M. & Ye, F. (2019). Meteorological Sensor Data Storage Mechanism Based on TimescaleDB and Kafka. *International Conference of Pioneering Computer Scientists, Engineers and Educators*, pp. 137–147.

Tampakis, P., Pelekis, N., Andrienko, N., Andrienko, G., Fuchs, G. & Theodoridis, Y. (2018). Time-aware sub-trajectory clustering in hermes@ postgresql. *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pp. 1581–1584.

Telecom. (2020, December, 2). Panasonic unveils 'smallest' pin-shaped lithium ion battery [html]. Retrieved from https://www.telecompaper.com/news/panasonic-unveils-smallest-pin-shaped-lithium-ion-battery--1041159.

Thomas, D., Wilkie, E. & Irvine, J. (2016). Comparison of Power Consumption of WiFi Inbuilt Internet of Things Device with Bluetooth Low Energy. *International Journal of Computer and Information Engineering*, 10(10), 1856–1859.

Tobola, A. et al. (2015). Sampling rate impact on energy consumption of biomedical signal processing systems. *2015 IEEE 12th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, pp. 1–6.

Tolba, A., Said, O. & Al-Makhadmeh, Z. (2019). MDS: Multi-level decision system for patient behavior analysis based on wearable device information. *Computer Communications*, 147, 180–187.

Townsend, K., Cufí, C., Davidson, R. et al. (2014). *Getting started with Bluetooth low energy: tools and techniques for low-power networking*. " O'Reilly Media, Inc.".

Valentin, G., Alcaidinho, J. & Jackson, M. M. (2015). The challenges of wearable computing for working dogs. *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers*, pp. 1279–1284.

Van Kranenburg, R. (2008). *The Internet of Things: A critique of ambient technology and the all-seeing network of RFID*. Institute of Network Cultures.

Varewyck, M. & Martens, J.-P. (2010). A practical approach to model selection for support vector machines with a Gaussian kernel. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(2), 330–340.

Viloria, A. et al. (2019). Integration of data mining techniques to PostgreSQL database manager system. *Procedia Computer Science*, 155, 575–580.

Weiss, G. et al. (2013). WagTag: a dog collar accessory for monitoring canine activity levels. *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*, pp. 405–414.

Wes McKinney. (2010). Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference*, pp. 56 - 61. doi: 10.25080/Majora-92bf1922-00a.

Xiao, H. & Eckert, C. (2013). Indicative support vector clustering with its application on anomaly detection. *2013 12th International Conference on Machine Learning and Applications*, 1, 273–276.

Yaman, S. & Pelecanos, J. (2013). Using polynomial kernel support vector machines for speaker verification. *IEEE Signal Processing Letters*, 20(9), 901–904.

Yang, Y. & Loog, M. (2018). A benchmark and comparison of active learning for logistic regression. *Pattern Recognition*, 83, 401–415.

Yonezawa, K., Miyaki, T. & Rekimoto, J. (2009). Cat@ Log: sensing device attachable to pet cats for supporting human-pet interaction. *Proceedings of the International Conference on Advances in Computer Enterntainment Technology*, pp. 149–156.

Zhang, Y. & Zhang, Z. (2019). Human Activity Recognition Based on Motion Sensor Using U-Net. *IEEE Access*, 7, 75213–75226.