THE EVALUATION OF RECOGNIZING AQUATIC ACTIVITIES THROUGH WEARABLE SENSORS AND MACHINE LEARNING

A Thesis

by

LARRY CLAYTON POWELL

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Tracy Hammond
Committee Members,	Daniel Goldberg
	Riccardo Bettati
Head of Department,	Dilma Da Silva

May 2019

Major Subject: Computer Engineering

ABSTRACT

Swimming is a complex and dangerous sport. A recent study found that swimming is the third leading cause of death among children in the world each year. A significant factor contributing to these statistics may be the limitations of current approaches to water-based education. As such, the Red Cross and Bangladesh have started investing in research into water-based education.

Current technology, monitors only the main swim styles backstroke, breaststroke, butterfly, and freestyle. These existing systems are missing additional activities, such as rest (treading water), transitions (flip turns), and low energy strokes (sidestroke). These additional activities have an effect on a person's swimming ability, and they form the baseline for what is taught by the Red Cross, Bangladesh, and the military.

We developed and tested an aqua-tracker system for monitoring swimmers in all forms of activities expected from a swimming-based training session. Our system uses a waterproof mobile device to capture a swimmer's flip-turns, ability to tread water, sidestroke, freestyle, backstroke, breaststroke, and butterfly strokes. Activities are recognized using a sliding-window framework, comparing both a deep learning and a feature-based recognition system. Our tracker has shown that the system can accurately detect each of the activities, from beginner to expert level, with an f-measure of .94. Equipped with the capabilities provided by our aqua-tracker system, people can monitor their own swimming ability, parents can monitor their children while they are in the water, and lifeguards and swimmers taking proficiency exams will be able to perform the exams without the needs of a proctor.

ACKNOWLEDGMENTS

I am grateful for the sketch recognition lab and all the members that provided support and motivated me to help with my master's thesis. I wish to express sincere gratitude to my advisor Dr. Tracy Hammond for her guidance in my work and support. I would also like to thank Dr. Daniel Goldberg and Dr. Riccardo Bettati for their insight in my work and with my master's thesis.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Dr. Tracy Hammond (advisor) of the Department of Computer Science & Engineering, Dr. Riccardo Bettati of the Department of Computer Science & Engineering, and Dr. Daniel Goldberg of the Department of Geography.

All work conducted for the thesis was completed by the student independently.

Funding Sources

This work was funded partly by the Texas A&M Engineering Experiment Station AggiE_Challenge course.

NOMENCLATURE

ML	Machine Learning
FFT	Fast Fourier Transform
MLP	Multi Layer Perceptron
CNN	Convolutional Neural Network

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

1. INTRODUCTION

1.1 Motivation

Swimming and water-related activities are fun and provide great exercise. Swimming is considered one of the best physical activities thanks to the buoyancy of water and the reduction of stress on physical joints. It allows everyone to be able to perform workouts and has been proven to affect all muscle groups, as well as increase lung capacity and improve breathing [1]. Furthermore, it can also help reduce weight gain and obesity. Swimming is a major part of society, with triathlons, Olympics, and many water-based sports promoting it to people of all ages and sizes.

While it is an extremely beneficial exercise, swimming can be very dangerous. Almost half a million people in the world die from drowning each year [2]. On average, ten people die every single day from drowning or other water-related accidents, and 1 of every 5 drowning victims is a child under the age of fourteen [3]. One of the major risks in these drowning-related deaths is poor swimming ability and lack of awareness of water dangers. This can be seen in the United States where more than 50% of Americans don't know how to swim, which increases to 61% for children [4, 5]. This can cause major neurological effects on children as they age, including mental stability and mental damage [6].

The current methods to counteract this situation are through coaching and teaching people multiple swim stroke techniques. The Red Cross suggests teaching basic swimming styles such as sidestroke, backstroke, breaststroke, freestyle, butterfly, and treading water. They have also created a set of exams in order to determine a person's swimming proficiency. These strokes and a person's proficiency to swim have been shown to help reduce the risk of drowning [7, 8].

While swimming courses and personal coaching can help reduce the risk of drowning, the number of drowning deaths continue to rise[1] [9]. This may be due to the high cost of swimming lessons, the difficulty of finding coaches who can provide rich, personalized feedback, or the lack of a standardized water-based education curriculum in the United States [10–13]. Technology may be able to provide a better, more accessible solution toward reducing the dangers of swimming, but current offerings are quite limited. We propose an aqua-tracking, wearable system which can detect a wide variety of swimming activities and provide feedback which may be used by coaches and trainers to more quickly provide feedback to a larger number of people, furthering our goal of making learning to swim an accessible skill to everyone.

1.2 Domain Context

There are many health and related areas that can be affected by wearable technology. Wearables allow users to track their health-related progress and understand if they are improving or not. Most current research has been focused on understanding physical activity in land-based environments.

The rise of mobile device technology has supported an increase in recognizing more types of physical activities including workouts, aerobics, sports, and many other activities. Recently, consumer devices like the Apple Watch have added capabilities to detect certain water-based activities [14]. When it comes to swimming there are four major stroke styles: freestyle, backstroke, breaststroke, and butterfly. These stroke styles are used in competitions, ranging from school events to professional and Olympic activities. For each of these competitions, flip-turns are used for transition-based swimming styles; this is when a person reverses their direction while swimming. These styles transition to multiple water-based events such as water polo and triathlons. For recreational swimming, there are a couple of minor styles that are important for safety, which are swimming techniques like

[1] https://sls.com.au/coastal-drowning-deaths-rise/

treading water and sidestroke.

1.3 Research Proposal

For this study, we targeted seven water-based activities. We covered the four major styles: breaststroke, butterfly, backstroke, and freestyle, along with three minor activities: flip turns, sidestroke, and treading water. We created software and hardware consisting of sensors, data storage, and a waterproof system to allow recognition of the seven swim styles. The developed algorithms can be separated into multiple different methods such as cleaning and understanding the sensor data. Developing, evaluating, and validating the features on multiple different time windows with several ML algorithms to find the best statistical measurement of the system. Finally, we developed a prototype system that can recognize and be tested on the user in a naturalistic setting.

The goal of this research is to answer the following questions:

1. What time and frequency domain features can accurately detect the four major swim strokes (backstroke, breaststroke, freestyle, butterfly) and the three minor strokes (sidestroke, treading water, flip turns)?

2. Is there a best sliding time window and machine learning algorithm that can detect all of the swim strokes?

3. Will a personal deep learning neural network provide better performance and optimized speed compared to other machine learning algorithms?

2. RELATED WORK

Research in activity recognition has provided promising results in many different fields from healthcare, military, and personal/commercial uses [15–20]. With the advancements in technology there have been expanding possibilities in using different sensors to recognize human activity. Research surrounding these systems has been focused on understanding sensors and sensor placement among activities. With the advancements in mobile development and increased computational performance from these devices, researchers have moved to recognize more and more human behaviors and activities. With the combination of sensor placement and activity, recognition algorithms can recognize many activities in real time. In industry, companies like Fitbit and Apple are using this technology to monitor health and physical activity and have focused on recognizing sleep, walking, running, and step counts [21, 22].

In the following sections we will cover work done in both land and water-based recognition systems and what activities are covered. We will also present an in-depth analysis of sensor placement and recognition methods that these systems use.

2.1 Wearable and Smart Phones

With the rise in products such as virtual realty, Apple, Samsung, and Google devices there have been an increase in senors types, computational power, and memory. With the rise of these ubiquitous smartphones, we have seen a rise in the simplistic capabilities of programming provided several advantages with it came to research and their capability [17]. The embedded system provides a new area of research with the concept of smartphones or applications [23–27] which have shown in the field of fashion, health, and physical activity that provide more insight and can validate ones capabilities. For non-embedded systems Kwapisz [16] used smart phones to recognize walking, jogging, ascending stairs, descending stairs, sitting, and standing with an accuracy of 91.7%. This has come to some trouble since two out of the six activities presented individually have no greater accuracy than 61%. A similar study used a phone-based accelerometer to detect walking, jogging, running, cycling, and sports which provided an accuracy of 71%. This is due to the fact that they claimed sports to have a variety of activities that overlapped the other classifications [28]. With water-based wearables we need to consider that limitations of water and its effect on these systems. This research [29–31] has shown that mobile devices can provide some capability in recognizing water-based activity. With MobyDick [31] showing the capabilities of developing a game for water-based swimming the system wasn't able to recognize the strokes in high accuracy and only relied on GPS signals. Marshall [29] used an android mobile device to gather sensor data from swimmers. This system was used to provide capabilities of mobile devices with water-based activities. The system did not provide a recognition algorithm or features but rather showed the understanding of if a mobile phone can gather sensor data in water-based activities.

2.2 Activity Recognition Wearable

Researchers have created many wearable solutions to recognize other human activities, including detecting and distinguishing daily care events such as brushing teeth, washing hands, or brushing hair [32]; detecting and distinguishing fitness activities including: jumping jacks, situps, pushups, and squats [33], detecting and distinguishing military emergency situations including: standing, lying down, and jumping [34]; and detecting and distinguishing pedestrian events including: walking, running, standing, sitting, or pacing [35]. Additionally, researchers at the Sketch Recognition Lab have created sensors to map conditions of the world around us including: ultraviolet rays [36] ozone levels [37], and even to aid in physical therapy [38], teach piano [39], and play an imaginary violin or cello [40].

These applications can be used in fields such as health care, assisted living, sports, learning environments, and security [41, 42]. Some of the earliest works in the field focused on identifying optimal sensor placement, as researchers demonstrated varying levels of success with sensors such as accelerometers, gyroscopes, and microphones commonly placed on various combinations of the wrists, upper and lower arms, waist, head, and chest [17, 43–47]. Pärkkä et al. [47] conducted a comprehensive study using nineteen different types of sensors to identify several common scenarios such as visiting the library, restaurants, or shops and common physical activities such as walking, running, and cycling. From this work, researchers found that accelerometers provided the most accurate indicator of what activity was being performed. Bao & Intille [48] tested biaxial accelerometers on the upper arm, wrist, hip, thigh, and ankle, and found that accelerometers placed on the thigh, hip, and ankle were the best indicators for activities that had some form of ambulation or posture, while accelerometers placed on the wrist and arm were better indicators for activities that involved mostly the upper body.

This research in features extraction and development of machine learning algorithms helps to detect activities. There has been researching in Deep Learning Neural networks. There are multiple types of neural networks from Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and Modular Neural Network. What we focus on is Multi-Layer Neural network which are classified as Feed Forward Networks which are easy to maintain and help reduce noise. Research [49–51] has focused on activity recognition systems use MLP as their machine learning recognition system. Another reason researchers use MLP is that they do not require many input nodes to provide the best performance comparatively to CNN. With the rise in features and new sensor systems there has been an increase in multiple using the other neural networks. Saez [52] presented that the use of a CNN is possible with 280 nodes as the input. These nodes were from six sensors: two accelerometers, one gyroscope, GPS, and one magnetometer each one provided a minimum of three features based on *x,y,z*-coordinates form each of the sensor systems. What they do not focus on is the power performance and real-time recognition of the system. CNN has also provided wide increase in activity recognition in medical health [53], physical activity [52], and genetics [54].

2.3 Sensor Placement

When it comes to sensor placement there has been much research on where to place sensors to recognize specific activities most accurately. Some research has focused on sensors placed on the chest [47, 55–57], wrist [58], lower back [59, 60], upper back [61, 62], upper arm [48, 57], head [63–65], ankle [56–58], hip [48, 55, 57], thigh [57, 66–68], and foot [17] for activity recognition. Cleland [69] presented the understanding of sensor placement and how it effects activity recognition. Bergman and Cleland research [69, 70] shows that lower back is a conventional location where it can improve compliance where it can be integrated in their daily life. Other research facilities created other forms of wearable systems to recognize and show capabilities of sensor systems correlation to swimmers techniques. Researchers presented that the location of sensors systems on the body can differentiate between freestyle, breaststroke, butterfly, and backstroke [71–77]. In the field of aquatic based recognition it is important to understanding sensor placement and numbers of sensors needed for optimal recognition. Current researchers has focused specifically on the wrists, lower back, upper back, chest, legs, and shoulders. With these studies their has been great results when it has come to sensor placement and research in recognition. A good example is Marc Bachlin [78] used two wrist sensors, an upper back sensor, and lower back sensors the purpose of the system is to compare mastery of the system this is done based on rotation, arm movement and angle of body. The use of multiple sensors was able to help the author with understanding that the important aspects of swimming is in the angle of body, hand motion, and locomotion. The author used this to recognize professional or elite swimmers.

2.4 Water Activities Recognition

There has been some other research in the field of water-based activity recognition. The main line of research is focused on recognizing the four major swim strokes (butterfly, backstroke, freestyle, and breaststroke). When it comes to this form of activity recognition, there are some important factors.

Sensor location is a major factor in activity recognition in general [57], as the location on the body for activities such as running, walking, and other land-based activities has a major effect on recognition precision. Swimming is no exception. Many researchers have explored the use of multiple sensor placements such as wrist, head, upper back, lower back, and ankle sensor placements [75–77]. Recent work has shown the lower back sensor placement is the current best method in recognizing swimming activity [79].

The other factor in recognizing water-based activities are the types of sensors being used such as gyroscope, accelerometer, and barometer. Choi [80] uses lower back sensor placement of barometer, accelerometer, and gyroscope to detect the swimming activities backcrawl (i.e., a type of backstroke), standing in the water, frontcrawl (i.e., freestyle), and breaststroke. Choi was able to recognize these four basic water activities at 96% precision through cross-validation. Our focus extends beyond these four activities to include the complete set of water activities used for proficiency exams and motions that help with reduction of drowning (e.g., treading water). The majority of papers only focus on a few of swim activities or try to focus on the activity recognition and not in real time [81–89]. We argue that there is a need to recognize the major swimming stroke activities needed for a proficient swimmer as well as the minor ones such as treading water and flip turns.

2.5 Differentiation

This research builds on the concept of single sensor placement. We decided the lower back is optimal with the current capabilities of a commercial mobile device. This allows for reduction of cost and simplicity of design. We include activities that have slight differentiation of movement and orientation of the body. The intention is to accurately and precisely detect the difference between nine swim strokes/activities (backstroke, breaststroke, freestyle, butterfly, sidestroke, treading water, and flip turns) in a swimming environment purely from the use of built-in sensors of a commercial mobile device (i.e., smartphone) in real-time.

3. STUDY DESIGN AND IMPLEMENTATION

The gathering of data and the methodology of the user studies influences the approach that can be used for labeling and analysis. We conducted user studies in a controlled and semi-naturalistic way to examine the activities of interest. In this chapter we describe the studies, system, and implementation of our experiments.

3.1 Study Goals

We had several goals in our study design to effectively measure the goodness and usability of our algorithm:

1. Collect enough data to measure the goodness of our algorithms.
2. Collect accurately labeled data.
3. Preserve some data for testing to ensure we do not over train on our available data.
4. Design a study that allows us to test if the algorithm could work in real-time.
5. Design a study that allows us to test if the algorithm works with both novice and expert users.

3.2 Obtaining Users

Obtaining a large amount of accurately-labeled swimming data can be a difficult prospect for a number of reasons: 1) not everyone knows how to swim, and not all participants are expert level swimmers, 2) performing studies at a pool requires permission from both the users and the pool administration, 3) obtaining an IRB for a swimming-based study combined with electronics is a long and tedious process due to the perceived risk of drowning.

In addition to getting the appropriate pool and IRB permissions, to ensure that we had enough accurately labeled good-quality data, we did two things: 1) designed a data collection app that allowed us and/or the swimmer to label the strokes in real time as we watched the swimmer (Section 3.4), and 2) worked with the TAMU swim-team (among others) to obtain a large number of users who would provide us with good-quality data (Section 3.3).

3.3 Gathering Participants

Obtaining participants with ranging swimming ability was a tough and complex process. Through the support of the Texas A&M Recreational Department we were able to communicate with several coaches and program directors that allowed us to gather varying skilled level participants. Through the department, we were able to work with the Master's program, Adult Beginner program, and Texas A&M Physical Education Activity Programs that are taught at the university.

3.4 Mobile Application Design and Implementation

To broaden the capabilities of a classification system, we conducted two types of user studies, one focused on recognizing specific stroke activities and another targeted at recognition in real-world scenarios. To collect accurately labelled data in both of these scenarios, we developed two Android smartphone applications. We chose the Android platform due to its open source platform and ability to easily support a wide amount of development constraints. Android supports a versatile and compatible development platform that is able to handle multiple tools and devices with backwards-compatibility. With the device's storage capability, our system is able to acquire sensor data for each participant as well as label the raw data for each swimming motion.

The mobile application supports the recording of a single swimming stroke (described in Section 3.4.1) as well as switching from stroke to stroke (described in Section 3.4.2).

Any method of recording allows the users name or other ID to be attached to the data.

3.4.1 Android Application 1: Interface to Record a Single Swimming Stroke

As part of our study, we requested users to swim a particular swimming stroke for a certain amount of time. To do this, the user would 1) open this application, 2) enter their Name or ID (if they hadn't already), 3) click the type of swimming activity (of six possible activities), 4) press "Start," 5) put the phone on their body (see Section 3.5), and then 6) perform the activity.

The phone records the accelerometer data for exactly 120 seconds only. This was for a variety of reasons: 1) The pressure of the water was interacting with the phone screen, making the Android application think that the user was pressing buttons. Thus, a "Stop" button was unfortunately impractical in this setting, usually being set up unintentionally. 2) This also ensured that the swimmers would stop after two minutes, allowing them scheduled in rest times, ensuring better quality data.

The phone was operated by the proctor instead of the swimmer until the swimmer became proficient with the app. Figure 3.1 is a visualization of the application that the proctor would use to conduct a study with this application.

Because the user has to place the phone into the satchel before swimming (and remove it after swimming), the first approximately 30–60 seconds of the activity were removed manually (Section 4) to ensure that the data collected is actually that of the person swimming.

3.4.2 Android Application 2: Interface Supporting Multiple Strokes

The first Android application (Section 3.4.1) only supports the data collection of a single stroke at a time. The user has to take off the device and reprogram it after each swimming stroke. This has two problems 1) the swimmer has to get out of the pool every 120 seconds, and 2) it does not allow us to collect longer data in a naturalistic setting.

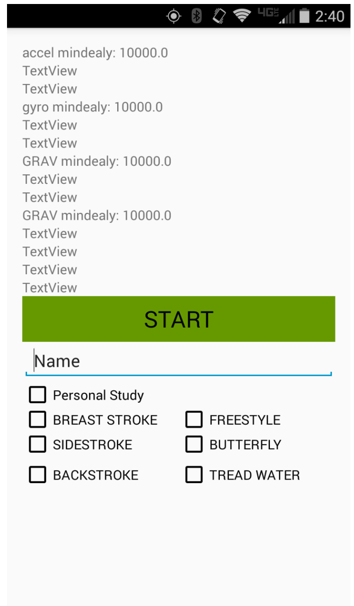


Figure 3.1: A screen shot of Android application created to measure a single stroke type at a time.

Since one of our main goals of this research is to create an app that can be used in a real-world setting and in real-time, we created a second Android application that allows the user to perform multiple types of strokes without having to take off the device. It had the disadvantage, however, of requiring two smartphones, one worn by the user and another administered by the proctor. Thus, we kept the original app in service, since that app was needed if the swimmer wanted to administer the phone. The single stroke application was still needed for any developmental testing, where the proctor and swimmer were the same person. The single stroke application was also helpful if the proctor was able to obtain multiple swimmers for testing at the same time.

As mentioned, the main advantage of the second application is to allow the the participant to never have to take off the device when swimming (so they don't have to come out of the pool every two minutes). However, it requires both a swimmer and a separate proctor. Figure 3.2b shows the application for both the swimmer and the proctor.

The interaction of the proctor and the swimmer with their perspective app is:

1. **Swimmer smartphone**: open the swimmer application and obtain the current participant number (essentially a count)

2. **Proctor smartphone**: open the proctor application

3. **Proctor smartphone**: press "Create Participant" on the proctor application and enter their participant number

4. **Swimmer smartphone**: press "Start,"

5. **Swimmer smartphone**: put the swimmer smartphone on their body (see Section 3.5)

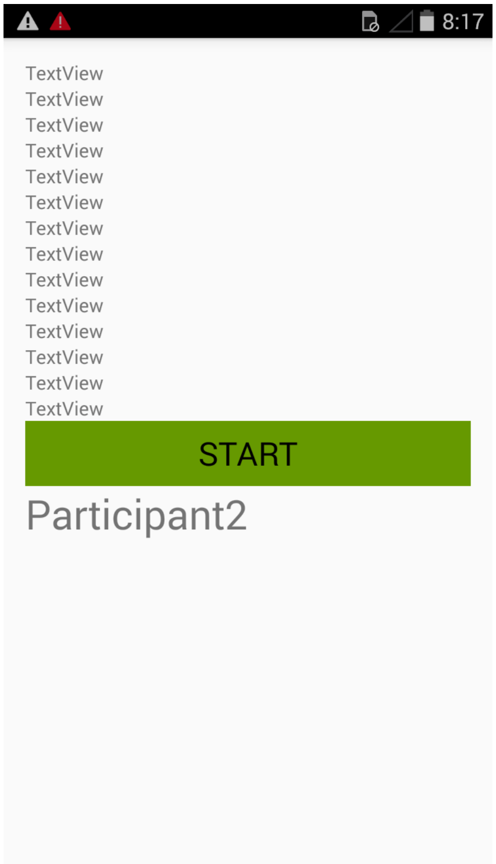
6. **Swimmer smartphone**: start swimming

7. **Proctor smartphone**: label activities on the proctor application as they occur by pressing the "Start" and "Stop" button next to each activity.

The two applications work in unison by flagging the data with the same participant number. In other words, to make use of each application's data, if the swimmer's app had "Participant 1" as its flag name, then the observer's application would also be required to have the same name.

As before, the swimmer's app runs only for a limited amount of time, but in this case it runs for 30 minutes instead of two minutes. 30 minutes was chosen based on feedback from the swimmers who used the first application.

One programmatic difficulty that we had to overcome was, when entering sleep mode, Android would deactivate data gathering after a short time. We solved this obstacle by activating a flag in the code to force the processor to keep running even when in sleep



(a) Swimmer's UI for Multiple Strokes Data Collection

(b) Proctor's UI for Multiple Strokes Data Collection

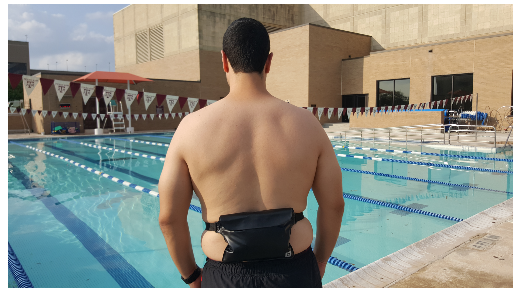
Figure 3.2: Screen shots of the Android application for collecting multiple strokes in a single swimming session.

mode. This drains battery significantly, so if the application is released, it would need to integrate a more efficient collection method.

3.5 Device Location and Water Proof Storage

We wanted to put the phone in a comfortable location that would not interrupt their swim pattern. We initially thought about wearing an arm cuff, but many of the expert swimmers complained that that would throw off their balance. Since they might only use the application for practice and not for competition, it is vital that their balance be the same

(a) Initial prototype design, involving a waterproof pack, an HTC Android phone, and harness



(b) Initial prototype design being worn by a swimmer

Figure 3.3: Version 1: Single-strap waterproof satchel

both with and without the device. Thus, rather than use a watch or arm cuff, we opted for a waist implementation. This obviously would have ramifications for the data collection as well, since accelerometer values will be very different when taken from the waist versus the arm or wrist. While many smartphones now boast that they are waterproof, we did not want to take that risk in this study. We developed a waterproof satchel that fits on the lower lining of the back.

3.5.1 Waterproof Satchel Version 1: Single-strap

Our preliminary design followed a similar approach to systems discussed in existing literature, with a single-strap system that fits the lower lining of the back [75]. Figure 3.3a shows the physical design of the system. Figure 3.3b depicts the system as it is expected to be worn by a swimmer.

3.5.2 Waterproof Satchel Version 2: Elastic Multi-Strap

In a round of pilot tests, we discovered that the initial single-strap satchel could be shifted by the water's movement over the course of a 30-minute session. We refined the design including additional straps and adding constraints to prevent the device from moving inside the waterproof enclosure. The new design moved from a single-strap to a

(a) Final prototype design

(b) Final prototype design being worn by a swimmer

Figure 3.4: Version 2: Elastic multi-strap waterproof satchel

3-strap system as seen in Figure 3.4a. Additionally, we now use elastic straps to allow the system to remain in close contact with the contours of the body. Finally, we added a second bag in which to hold the device. This allowed for improved ease of setup for user studies since the pack fit perfectly around the participant's back 3.4b.

3.6 User Study Design

We performed two separate user studies. The first study used the Android application described in Section 3.4.1 with the goal of creating accurate user data for a single stroke. The second study used the second Android application described in Section 3.4.2 with the goal of obtaining data in a more naturalistic setting. This section describes the strokes collected for each.

3.6.1 User Study 1: Single Stroke Data Collection

Fifteen users (six females, nine males) participated in the single stroke user study. The participants ranged in expertise from beginner to advanced proficiency in swimming. The participants met the study proctor at the pool where: 1) the user filled out a pre-questionnaire on their competency of swimming ability and the swimming styles they are able to perform. 2) the user put on the satchel, and the proctor tested the smartphone application first on land to ensure that everything was working correctly. 3) the participants were then asked to swim two laps of each different type of swimming activity that they were comfortable with in a calm body of water (pool). 4) the user completed a post-questionnaire and feedback on the system. More details of each step are given below.

3.6.1.1 Pre-questionnaire

For the pre-questionnaire, we gathered the following information:

1. What is your age?
2. What is your gender?
3. What swimming strokes do you know how to perform (freestyle, backstroke, breaststroke, butterfly, sidestroke, treading water)?
4. Form previous question: list from **best** (LEFT) **to worst** (RIGHT) the strokes you feel you have the most experience in.
5. How experienced a swimmer are you? **1 (Least) to 10 (Experienced)**
6. Have you ever performed in a wearable user study yes/no?
7. If yes, what was the study about?

The goal of the pre-questionnaire was to gather information that supplements the data. For instance, we wished to provide more insight into how the swimming style can be presented as well as find features correlating with proficiency levels.

3.6.1.2 Device Testing

After the pre-questionnaire was completed, we fitted the swimmer with the satchel, adjusting the satchel with the phone inserted until it felt natural and the user was able to swim comfortably with it. The proctor then started the application and asked the user to walk a few paces on land to make sure that the device was accurately collecting data. Once the device was confirmed as working, the study progressed to the swimming stage.

3.6.1.3 Swimming

The participants were then asked to swim the styles that they stated they were able to perform based on their pre-questionnaire. For each swimming activity, the user was asked to swim 50 meters (two laps of a 25 meter pool). Once they completed each swimming activity, they were instructed to leave the pool and sit back on the side of the pool to take off the device. They would repeat these actions until all swim styles were completed. This provided us with with a total of two laps of data for each swimming activity.

3.6.1.4 Post-questionnaire

The post-questionnaire reflected on the position of the device and feedback on the data collection mechanism.

1. Was the haptic (vibration) feedback responsive and intuitive?

2. Was the wearable device noticeable while swimming? If so, why?

3. Where would you prefer the device to be if you to use it for physical activity?

4. List the most negative aspect(s) of using the device?
5. List the most positive aspect(s) of using the device?

These questions give insight into how people react to a wearable swimming device and, in particular, their comfort with the vibrations and forms of haptic feedback. They also provided aspects of the wearable device which could be improved.

3.6.2 User Study 2: Multiple Strokes in a Single Data Collection

The goal of our second study was to collect data in a semi-naturalistic setting, i.e., we did not want the user to have to exit the pool after every swimming activity and reset the device. Using two smartphones, one for the proctor and one for the swimmer as described in Section 3.4.2, we gathered continuous swimming data for five participants. Because of TAMU IRB delays in approval, we were not given permission in time by TAMU IRB to collect demographic information for this second group of users (only the accelerometer data itself). Thus, we do not know the gender nor experience level of this second group of users.

In this study, the users were again first set up to ensure device comfort and that the data collection applications were working properly. Each user was then asked to swim for a total of thirty minutes, performing any of the swimming activities that they were comfortable with in the allotted time. A proctor, with experience in all of the swimming techniques, observed them while they swam. The proctor labelled the data in real-time, pressing “Start” and “Stop” for the appropriate swim type on the proctor’s smartphone application. These labels were applied to the continuous data stream later during analysis. This second study was conducted with five participants with the majority providing approximately thirty minutes of swimming data.

4. PROCESSING AND MODIFYING RAW DATA

Raw data from both user studies contains multiple activities per file stored, which needs to be cleaned and re-labelled. For example, in User Study 1, we need to separate the pre-swimming activities from the swimming activities. Cleaning of data required a process to accurately determine specific activities and separating it from the raw data. We developed a system to accomplish these tasks for both sets of data related to each user study. In this chapter we examine the methods taken in visualizing and cleaning of the raw data.

4.1 Data File Representation and Visualization

In User Study 1, the user only performs a single stroke type, but we need to separate the non-swimming activities from the swimming activities. In User Study 2, we have time labels to help us automatically distinguish and separate different stroke times in the data, however, we still must manually examine the data to make certain that there are no errors in the labeling.

In order to manually divide the data accurately, we first need to visualize the raw data. Visualization of raw data is necessary to see patterns and separate the activities that are in each file. We use the python library *pyplt* to visualize the data with a graph. These graphs, displaying the x,y,z, time raw data, provide a feature allowing zooming for precision data division. Figure 4.1 shows sample data from User Study 1 zoomed in and zoomed out, allowing us to manually accurately divide the pre-swimming activity from the swimming activity. Figure 4.2 shows sample data from User Study 2 zoomed in and out complete with the labeled time stamps. By zooming in and out we can ensure that the data is divided accurately for training purposes.

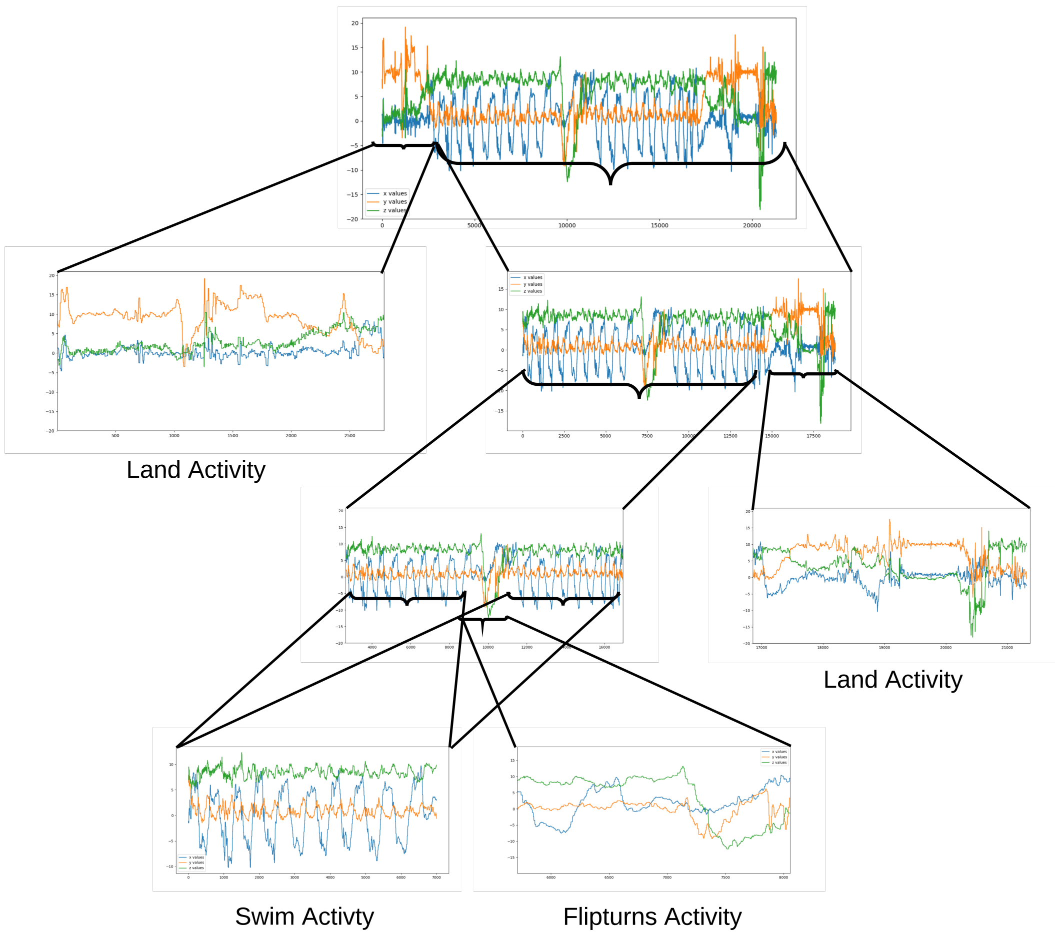


Figure 4.1: Visual method to processing data from User Study 1

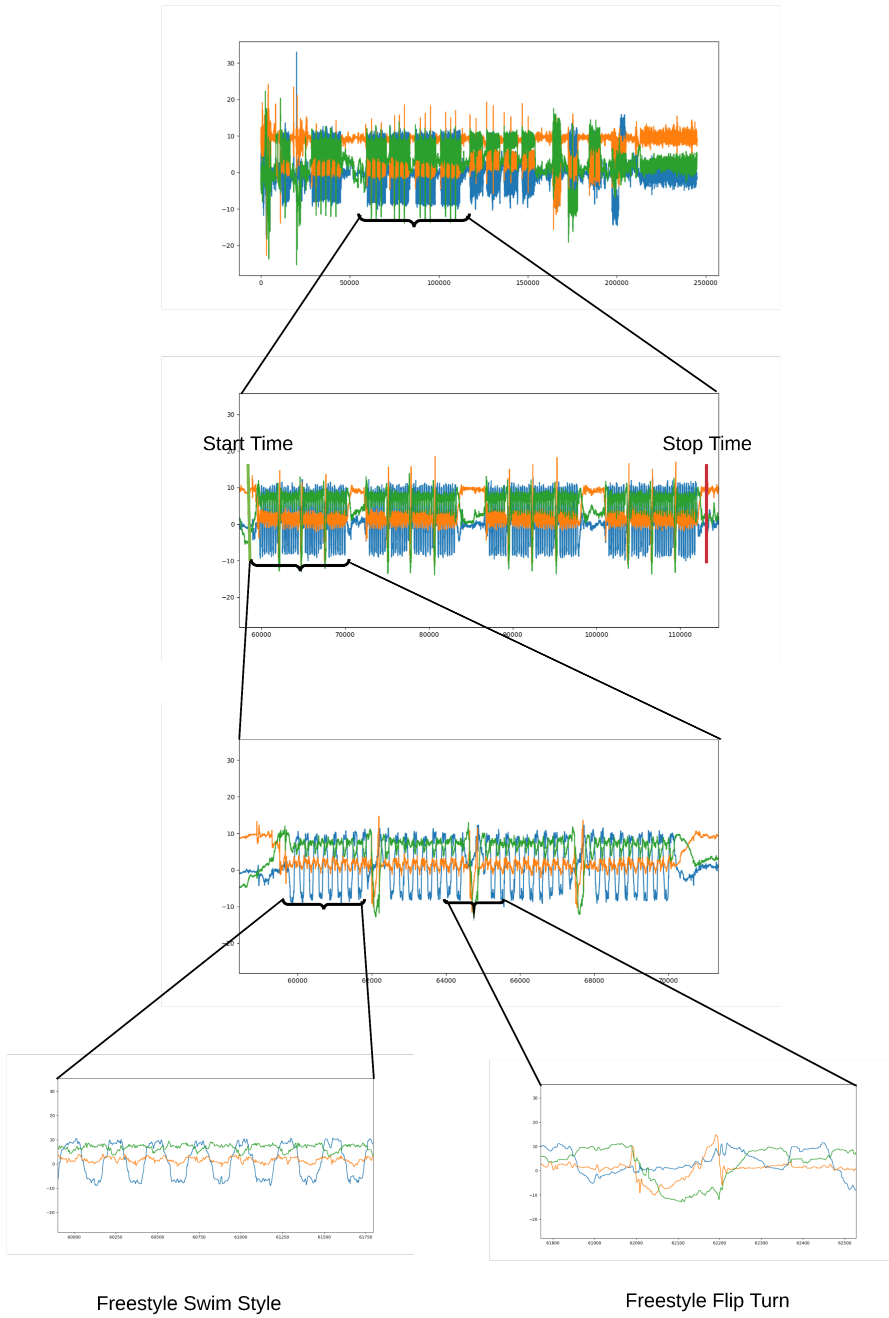


Figure 4.2: Visual method to processing data from User Study 2

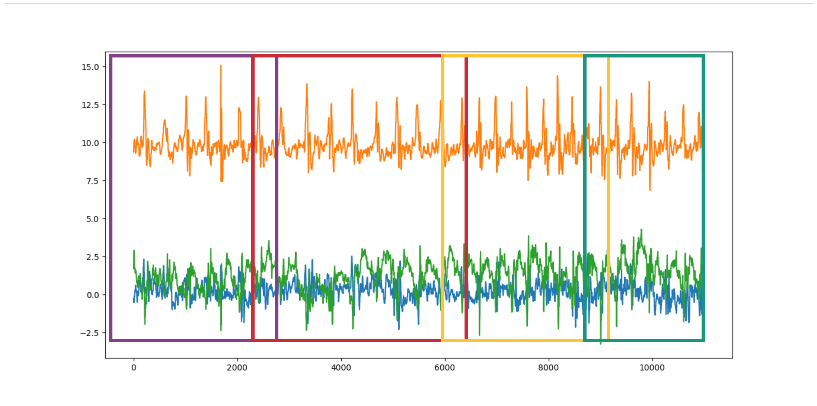


Figure 4.3: (X = Blue, Y = Orange, Z = Green) Sliding window representation based on the colored blocks overlapping with the raw data

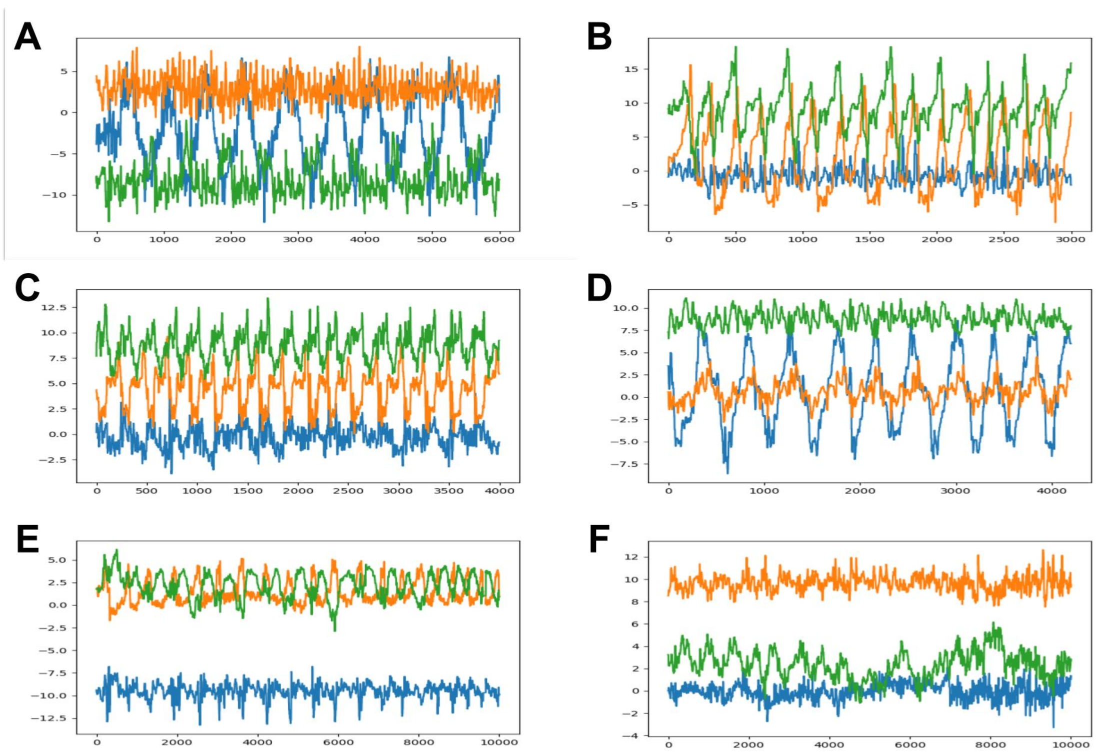


Figure 4.4: (X = Blue, Y = Orange, Z = Green) Sample graph of each activity where A = freestyle, B = butterfly, C = breaststroke, D = backstroke, E = sidestroke, and F = treading water

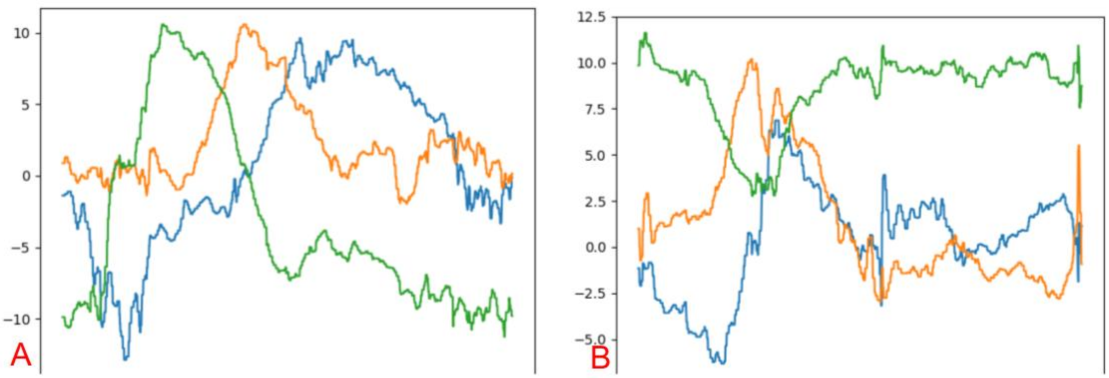


Figure 4.5: (X = Blue, Y = Orange, Z = Green) These are Sample graphs of each activity where A = Backstroke/Freestyle Flip Turn, B = Breaststroke/Butterfly Flip Turn

4.2 Sliding Window

We recognize activities by examining time segments for classification with the use of a sliding window shown in Figure 4.3. The length of a window is based on how long it takes to perform the specific activity. For example, the total expected time of brushing one's teeth takes two minutes, and four-second windows have been shown successful to identify partial moments of activity [90]. The sliding windows, shift the time segments specified for recognition. As shown in Figure 4.3, when we shift the window we overlap some of the data to help ensure that the core of the activity is centered on some window and better support activity transitions. To determine the best sliding window for water based activity recognition we examined swimming patterns, looking at how long a single arm cyle motion is for each style. A recent paper [91] suggested that 2.5 second windows was ideal for distinguishing between freestyle, backstroke, and breaststroke. Using 2.5-second windows as a baseline time for our system, we tested our recognition on 1.5, 2, 2.5, 3, 3.5, and 4-second windows with an overlap of 250 to 500 milliseconds. The purpose for testing additional times is to determine which window size is best when we also include additional swimming strokes (flip turns, treading water, sidestroke).

4.3 User Study 1 Data Processing

In User Study 1, each swimming stroke is recorded in a separate file containing 120 seconds worth of data (Section 4.1). Due to varying skill levels, the participants completed two-laps of the pool in varying times. The system starts recording at when the swimmer is still on land, and in addition to the swimming stroke itself, will also include a flip turn at the end of the first lap. We visualized the data through the python graph (Section 4.1), then took several steps to precisely separate the data to their corresponding file (land, specific stroke data, or flip turn). Visualization of these steps are presented in Figure 4.1 and on the list below.

1. Produce raw data in a graph
2. Examine and find "starting location" of a pattern in the graph
3. Remove all data up to "starting location" and reproduce the graph
4. Find "ending location" where pattern ends
5. Remove all data after "end location" and reproduce the graph of a single activity type
6. Write data to a text file with the name being the classification

4.4 User Study 2 Data Processing

Processing of User Study 2's data is more complex due to the amount, or rather types, of data in a single file. Multiple segmentation need to happen to implementation of a second segmentation method was needed for the raw data. The purpose for this second segmentation is due to the swimmer's raw data consisting of many different strokes stored in a single file. Each stroke type needs to be matched to a timestamp, label, and start/end stored in a separate file. For example in Figure 4.2 shows where a start and end timestamp in represented for a swimming style, with the steps listed below:

1. Produce raw data between start and end timestamp
2. Examine and find “starting location” of a pattern in the graph
3. Remove all data up to “starting location” and reproduce the graph
4. Find “ending location” where pattern ends
5. Remove all data after “end location” and reproduce the graph
6. Write data between both locations to a text file with the name being the classification
7. Remove data up to ”end location” from larger raw data
8. Repeat step 2 to step 8 until all activities between start and end timestamp are completed

5. FEATURE EXTRACTION

Activity recognition relies on features to distinguish activities from each other. We analyze the data and developed novel features and compare them to the traditional features used in previous papers. In this chapter, we discuss the mathematical algorithms of the traditional features and the novel features developed.

5.1 Traditional Features

Existing studies [79, 91] show a distinct set of features that provided high accuracy when detecting swim strokes. These features used in previous papers we will classify as traditional features. In the papers the traditional features were grouped between two general categories: 1) Time Domain and 2) Frequency Domain.

5.1.1 Traditional Time Domain Features

When addressing the time domain features the focus is on the value set within that window of time (Section 4.2). We address these as algorithms that use the entire array of data sets. The flaw with time domain features is the inability to focus on a piece within the pattern to pull important classifications. In the list we present below is the formulas and algorithms used as traditional features. Each of these nine features were computed from the three dimensions of accelerometer data (x, y, and z axis), producing a total of 27 features used for activity recognition.

- Peak count (x,y,z)

$$\begin{aligned} peak\_value\_threshold = median(a)+ \\ ((median(a) - max(a)) * percent\_threshold) \end{aligned} \tag{5.1}$$

- Valley count (x,y,z)

$$\begin{aligned} valley\_value\_threshold = median(a) - \\ ((median(a) - min(a)) * percent\_threshold) \end{aligned} \tag{5.2}$$

- Mean (x,y,z)

$$\bar{a} = \frac{\sum a}{n} \tag{5.3}$$

- Standard deviation (x,y,z)

$$\sigma = \sqrt{\frac{\sum (a-\bar{a})^2}{n-1}} \tag{5.4}$$

- Root mean square (x,y,z)

$$a_{rms} = \sqrt{\frac{\sum a^2}{n}} \tag{5.5}$$

- Skewness (x,y,z)

$$\gamma = \frac{\sum \frac{(a-\bar{a})^3}{n}}{\sigma^3} \tag{5.6}$$

- Kurtosis (x,y,z)

$$K = \frac{\sum \frac{(a-\bar{a})^4}{n}}{\sigma^4} \tag{5.7}$$

- Correlation coefficient (xy,yz,xz)

$$CorrelationCoefficient_{(x,y)} = \frac{cov(x,y))}{\sigma_x \sigma_y} \tag{5.8}$$

Peak and valley threshold features are used to distinguish patterns for counting the amount of peaks and valleys from the data sets. The way we use the performance of the thresholds is to provide flags among the data within a time window. The steps taken when counting the amount of peaks and valleys can be presented in the pseudocodes 5.1, with a visualization of the algorithms provided by Figure 5.1.

Table 5.1: Peak and Valley Algorithms In Python

```
def findpeakscount(data_array,percent_limit):
    maxgroup = (max(data_array) - median(data_array)) * percent_limit
    maxgroup = median(data_array) + maxgroup

    counting = 0
    in_peak_mode = False
    for x in data_array:
        if(x >= maxgroup):
            in_peak_mode = True

        if(x <= maxgroup) and in_peak_mode:
            counting+=1
            in_peak_mode = False
    return counting
```

```
def findvalleyscount(data_array,percent_limit):
    maxgroup = (max(data_array) - median(data_array)) * percent_limit
    maxgroup = median(data_array) - maxgroup

    counting = 0
    in_valley_mode = False
    for x in data_array:
        if(x >= maxgroup):
            in_valley_mode = True

        if(x <= maxgroup) and in_valley_mode:
            counting+=1
            in_valley_mode = False
    return counting
```

5.1.2 Traditional Frequency Domain Features

Frequency Domain feature category is a series of algorithms where the features are extracted based on the repetition correlation of the periodic nature. These features use frequency components such as Fast Fourier Transformation (FFT) or Fast Time-Frequency Transformation (FTFT). The only feature that we use in this category is Domain Frequency

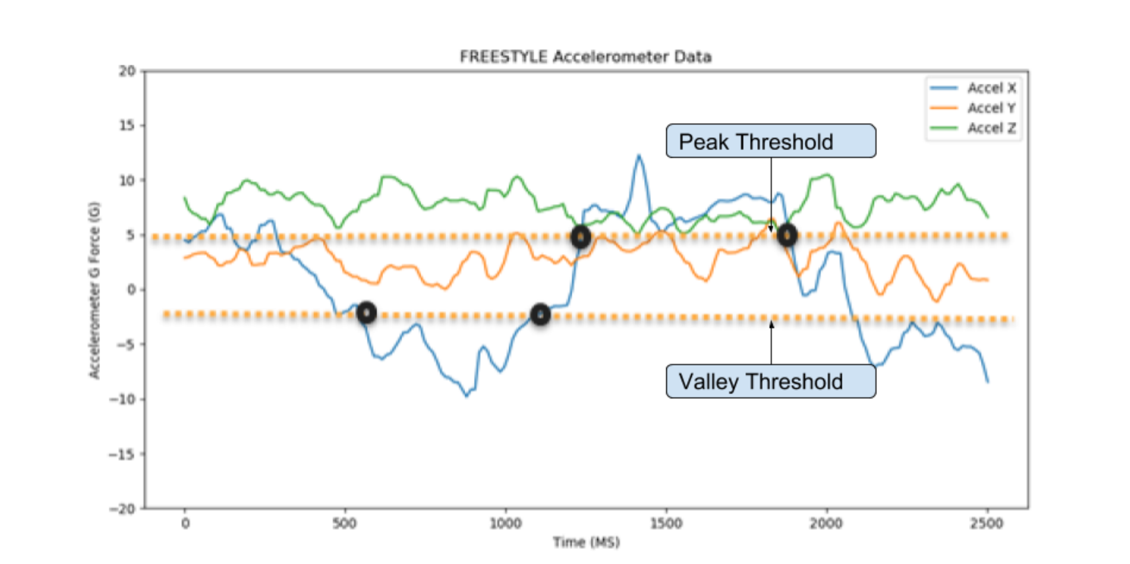


Figure 5.1: Example peak and valley threshold for a time window

Entropy of the accelerometer data for motion in the X, Y, and Z axes. Many papers such as [48, 92] present the methods in extracting information gain or Frequency Domain Entropy. This can be shown in Formula 5.9.

$$FrequencyDomainEntropy = \frac{\sqrt{a_i^2+b_i^2}}{\sum_{k=0}^{N-1}\sqrt{a_i^2+b_i^2}}$$
$$a_i = x_i cos(\frac{2\pi f_i}{N})\, and\, b_i = x_i sin(\frac{2\pi f_i}{N}) \tag{5.9}$$

5.2 Novel Features

Breathing patterns and swim stroke motions vary among swimmers based on size and expertise. There is a need to focus on distinguished swim strokes based on a single body motion. The traditional features really on the entire array of swimming data per window. This issue causes butterfly and breast stroke as well flip turns to produce inaccuracies. We derived our novel features based on the peaks and valleys that are in the varying time sliding window sizes. We use peaks and valleys because they correlate to the locomotion and breathing patterns of the swim styles. We focused on these swim styles' breathing patterns due to each person having several varying breathing intervals, and with the time window, the swim style's pattern count would vary upon each individual.

5.2.1 Time Domain Novel Features

With motions of the swimmers having a pattern and fluctuations in the data. We developed the time domain features focus on enhancing these fluctuation. The first features we implemented cross correlation equation 5.10 to distinguish the displacement of the accelerometer axis to each other.

$$CrossCorrelation_{(x,y)} = max_{d=1}^{n-1}(\frac{1}{n}\sum_{i=1}^{n} x_i \cdot y_{i-d}) \tag{5.10}$$

As we examined the data there is differences in the value range between stroke types. Zero crossing feature provides a middle point in the range for counting the number of times its passed. The Figure 5.2 shows the representation of how the feature works on an example data set.

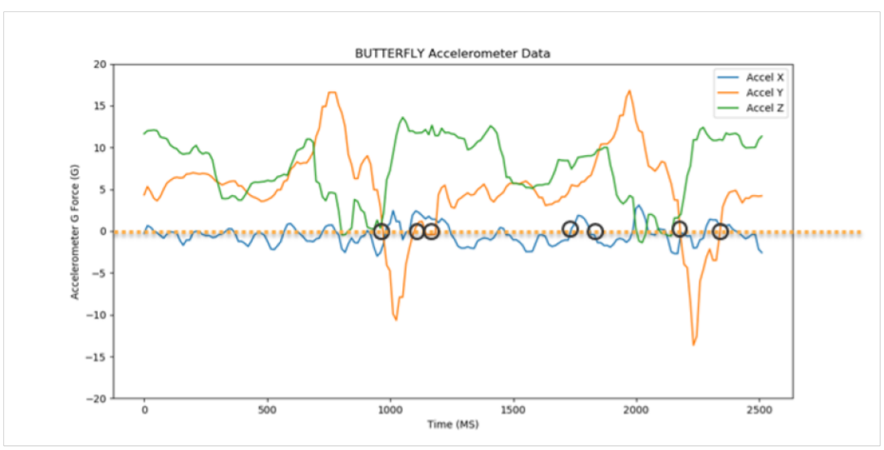
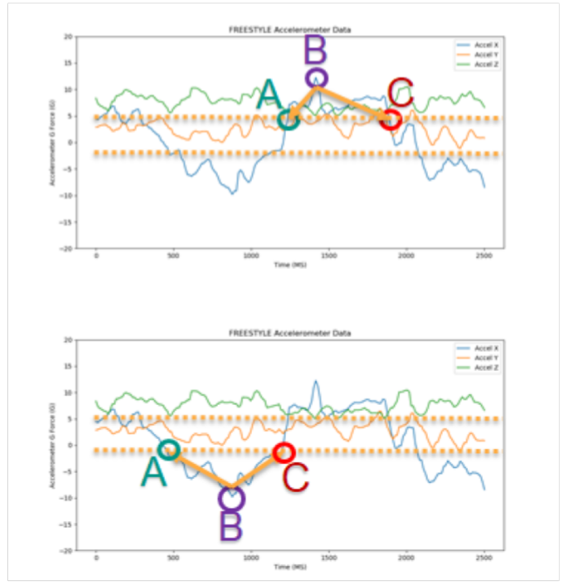


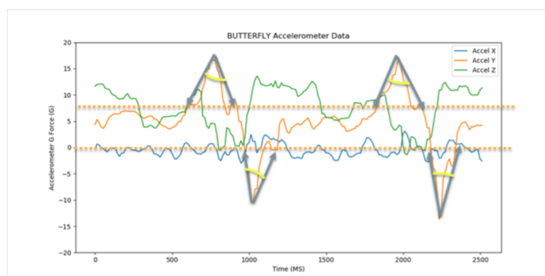
Figure 5.2: Example zero crossing for a time window

The peak and valley features in the data represent the motions of the swimmer breathing or performing a stroke. Focusing on the development of several features related to the size, length of the peak and valley set. To address this, we use the angle from the max peak value or min valley value to the passing of the thresholds. Figure 5.2.1 demonstrates how we acquire A, B, and C point for the equation 5.11. Within the time window, there will be several peak and values which are represented in Figure 5.2.1. Ultimately, we used the total angled values to pull several features shown in the list 5.2.1.

$$A^\circ = cos^{-1}\left(\frac{ba \cdot bc}{2ba * bc}\right) \tag{5.11}$$

- max peak angle x,y,z
- min peak angle x,y,z
- average peak angle x,y,z
- max valley angle x,y,z
- min valley angle x,y,z
- average valley angle x,y,z
- total angle average x,y,z





5.2.2 Frequency Domain Novel Features

For frequency-domain features, we use similarity in the form of a Fast Fourier Transformation (Equation 5.9), which gathers varying frequency values for the entire data set. We focus on increasing said values and calculating the power spectral density (Equation 5.13) to measure the power of the signal compared to the usual frequency. The DC component (Equation 5.12) is used to measure the Discrete Frequency of the signal based on 0 Hz.

$$DC = \frac{1}{N} \sum_{i=0}^{N-1} a_i^2 \tag{5.12}$$

$$PowerSeptralDensity = \frac{1}{N} \sum_{i=0}^{N-1} a_i^2 + b_i^2 \tag{5.13}$$

6. MACHINE LEARNING EVALUATION AND METHODOLOGY

Machine learning is the prediction of outcomes based on a set of algorithms. For activity recognition machine learning can classify and detect activities based on a feature set. We have presented what features (Chapter 5) we have extracted from the swimming styles. We have to focus on how we plan to evaluate these feature sets and compare the novel and traditional features. In this chapter, we present: 1) the filtering of featured data sets; 2) the machine learning algorithms tested; 3) the validation methods used to determine the best algorithms and feature set; 4) the deep learning neural network layout and development.

6.1 Filtering Method

With the large amount of features there may be some that have no correlation to the activities or help in classifying them. The way we evaluate those best is through a filtering method. The most popular one is best-first subset selection. This filtering method uses forward evaluation on the paths of sub features that have the highest correlation when distinguishing between classifications. For our analysis we use subset selection on the entire training data set of fifteen participants. To avoid over fitting of the data, we 1) use leave-one-user-out independent validation for subset selection, and 2) later test our reduced set of features again on a separate new set of test data with different participants. The goal of this filtering step is to provide features that distinguish between all activities.

6.2 Machine Learning Algorithms

From the wide range of machine learning classification algorithms, we decided to focus on the Multilayer Perceptron, Naive Bayes, Random Forest, Random Tree, J48 Decision Tree, Decision Stump, Nearest Neighbor, and RepTree classifiers. This selection gives us a diverse set of the most popular algorithms.

The Multilayer Perceptron is a neural network where the first layer of nodes is connected to each feature. From there the first layer is connected to another layer. That layer is called a hidden layer where each connection has a weighted value. The hidden layer is then connected to a final node where, if the final value is closer to one, it classifies it based on what that node is. Equation 6.1 shows the described multilayer. Neural networks tend to perform quite well, but the multilayers tend to obfuscate the inner-workings of the network.

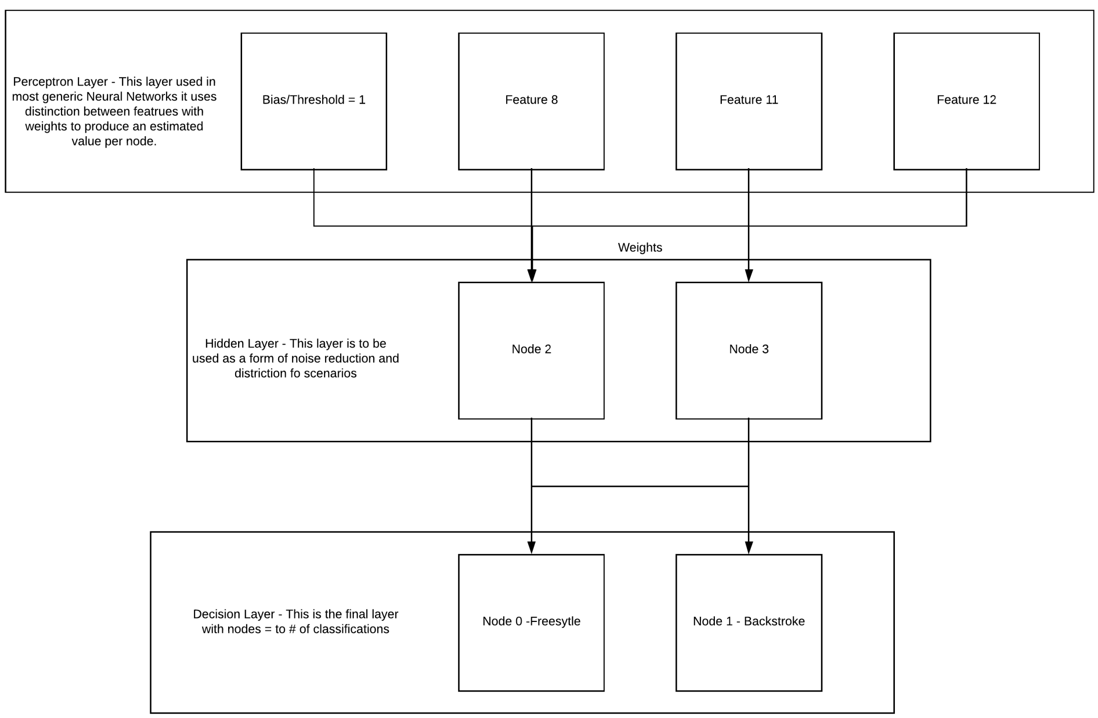


Figure 6.1: Sample of what a ML perceptron looks like

Decision trees are trees with if-else statements based on values determined from a correlation algorithm. There are many kinds of decision trees: Random Forest, Random Tree, J48 Decision Tree, Decision Stump, and REPTree. Random Forest involves many decision trees that decides a classification based on individual tree decisions and produces a confidence value, returning the classification that has the highest such value. The J48 decision tree is the fastest and most basic tree, which uses information entropy and information gain as the algorithms to build the tree. Random Tree uses a random feature as a baseline and builds based on information gain from that starting node. REPTree uses regression from information gain and prunes the tree with reduced-error pruning. Finally, Decision Stump is a single level decision tree where the root is connected to the classification. The decision tree, and by that nature also the decision stump, are useful because they produce a human-readable algorithm of rules of if/then statements, from which we can try to better understand how and why certain features are useful. A decision tree is also very easy to implement in any mobile phone or microprocessor and requires very little processing power, time, or space.

The next algorithm is the Naive Bayes classifier, which is a probabilistic classifier that uses probability function from Gaussian Distribution to distinguish what feature goes to what classification. By its very nature, a Bayes classifier is good at taking prior probabilities into account.

Last is the nearest neighbor which has a long run time and requires more space compared the other machine learning algorithms. The algorithm uses a distance algorithm such as Euclidean distance for each feature from the guessing input and all the classifications that are in the data. This is slow since it compares the input to each classification example. After that, it takes the classification with the closes distance and claims be the output for the data input. This is a suitable algorithm for large data sets since it can also help reduce noise. It is particularly good when members of the same class may look more differently from each other than they do to members of different classes. Its speed, however, limits its efficiency in modern applications.

6.3 Training Data Validation

To determine the best algorithm that will work we use an independent validation which is when the data is separated based on the number of participants.This method is better than the current method of cross validation which is when the total data set is randomly grouped based on the number of folds. This is bad since some of the data from one participant may end up in both training and testing. The accuracy would be to high due to the machine learning algorithm fitting to the training data. The independent validation will reduce the risk of over-fitting the algorithms and producing too high of results.

For deciding the statistical values of the ML algorithms, we have the true positives (TP), false positives (FP), and false negatives (FN) and use them for calculation of multiple statistic values such as recall (Equation 6.2), sensitivity, precision (Equation 6.1), f-measure, and accuracy. Among the statistic values, f-measure is the most important since it determines are values based on Equation 6.3. The statistical algorithms also consider the amount of each classification, since there might be an uneven distribution in 1 classification over-encumbering the data compared to another.

$$precision = \frac{TP}{TP + FP} \tag{6.1}$$

$$recall = \frac{TP}{TP + FN} \tag{6.2}$$

$$f\text{-}measure = 2 * \frac{precision * recall}{precision + recall} \tag{6.3}$$

6.4 Testing Data Validation

Our goal in testing was to determine which features best approximate the different swim strokes and swim styles. To prevent over-fitting, the data was a) randomly separated into 1ten different data groups to allow for ten-fold cross validation and b) manually separated into groups containing a single user's data in each group to allow for leave-one-out cross-fold validation. We first trained and tested on the six swim styles (freestyle, back-stroke, butterfly, backstroke, treading water, and sidestroke) using 10-fold cross validation on several sliding window sizes. We performed three different scenarios on these swim stroke styles. With each scenario using cross and then leave-one-user-out user independent validation. For the scenarios is was to use the original and current features as presented in Section 2 and Section 3. We then use the new novel features with the final scenario being the combination of the novel and original features. For all these scenarios we performed them on multiple sliding windows of 2000, 2500, 3000, 3500, and 4000 milliseconds with a 500 and 250-millisecond overlap.

We then used the same methodology for the flip turns, although the sliding windows needed to be changed due to flip turns taking less time to perform. We chose to use 1500, 2000, 2500 millisecond sliding window with 500 and 250-millisecond overlap. For flip turns professionals have two classifications backstroke\freestyle flip turns and breaststroke\butterfly flip turns.

Finally, we take all the six major strokes and flip turns to develop a ML algorithm that can recognize the total stroke types. We then test on the eight swim styles through the same sliding windows (1500, 2000, 2500) as the flip turns since we want the time intervals to be similar for the real-time recognition. We compared all these statistics values and determine the best fitting time window, feature set, and machine learing algorithm.

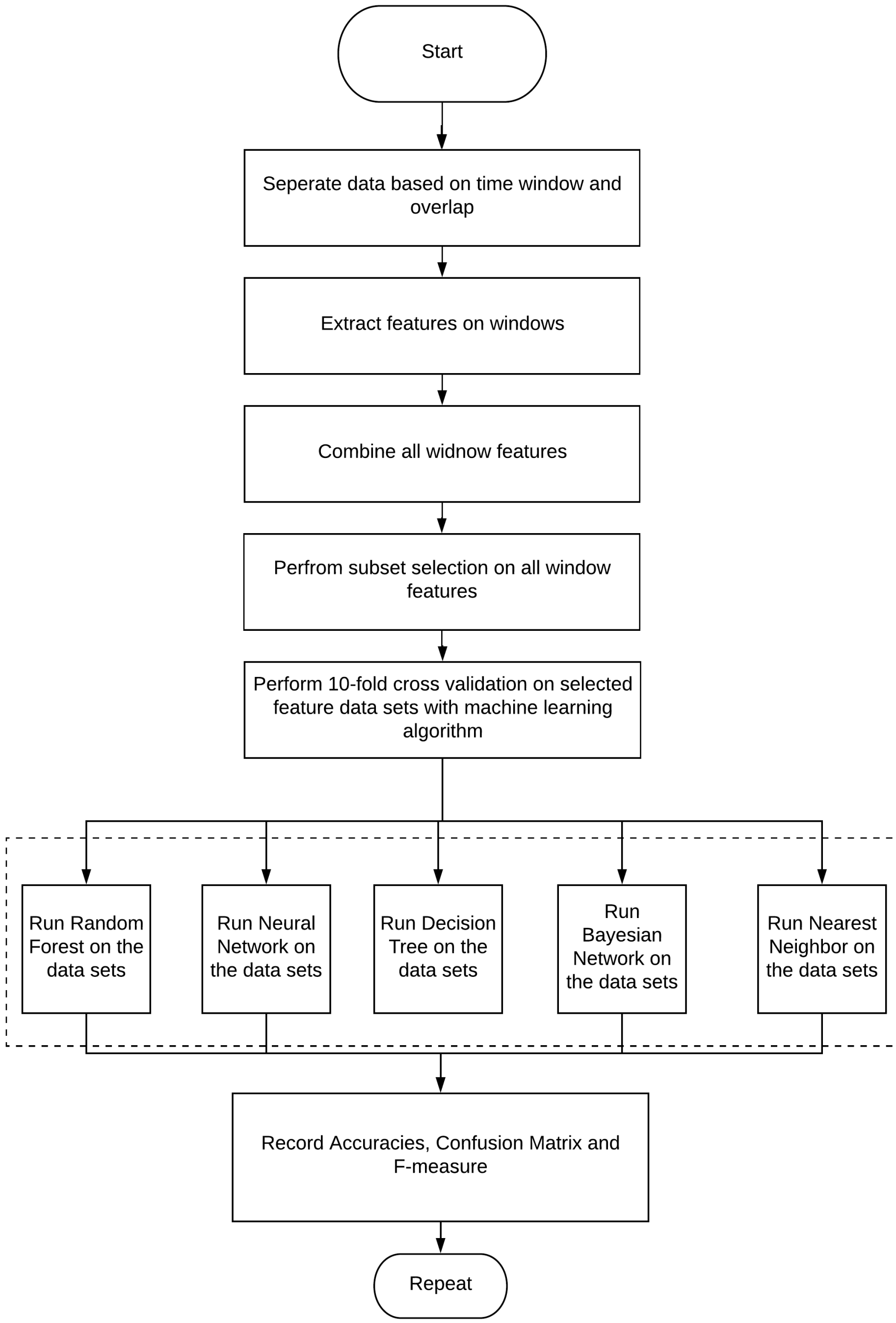


Figure 6.2: Testing Layout for ML algorithms

7. DEEP LEARNING EVALUATION AND METHODOLOGY

With the advancements of deep learning and its capability performing tasks quickly and efficiently, we propose a personal deep learning system to recognize swimming activity. This chapter describes the methodology in the development of a deep learning system and the processing of input data.

7.1 Resampling Data

Visualization of the raw data identified that the application doesn't constantly gather sensor data at 100 Hz. The way we overcame this issue is through re-sampling. Re-sampling helps us overcome latency that exists during value updates for the device. Re-sampling allows us to specify the number of data points we desire per sliding window (Section 4.2). The re-sampling algorithm is shown in the pseudocode Algorithm 1. Currently, the deep learning system uses the re-sampling method to modify the dataset to allow for the data to be input into the convolutional layer of the neural network.

7.2 Neural Network Data Modification

Features extracted details the patterns and thresholds that differ between swimming styles. For our deep learning neural network, we focus on the capabilities of using the raw data set as the form of input. Our system uses a convolutional neural network which requires the data to be in a 2-dimensional array. The way we were able to overcome that is that we re-sampled the data in the sliding windows. We would then take that re-sampled data and convert it into a 2D array. This is a tough process because the data being used is on three axes. That means the array needs to be able to handle three data groupings. The way to overcome all these mathematical issues to determine the best convolutional input is to find a square that is divisible by 3 * 2 * (# of convolutional layers). After that

Algorithm 1 Time based re-sampling

```
def distance(p1,p2):
    dx = p2[0] - p1[0]
    dy = p2[1] - p1[1]
    return float(math.sqrt((dx*dx) + (dy*dy)))

def pathlength(points):
    d = 0.0
    for index in range(1,len(points)):
        d += distance(points[index-1],points[index])
    return float(d)

def resample(points, totalPoints):
    I = pathlength(points) / (totalPoints - 1)
    D = 0.0
    newpoints = [points[0]]
    i = 1
    while i <= len(points)-1:
        d = distance(points[i-1],points[i])
        if((D+d) >= I):
            qx = points[i-1][0] + ((I-D)/d)*(points[i][0]-points[i-1][0
                                                        ])
            qy = points[i-1][1] + ((I-D)/d)*(points[i][1]-points[i-1][1
                                                        ])
            q = [qx,qy]
            newpoints.append(q)
            points.insert(i,q)
            D = 0.0
        else:
            D += d
        i+=1
    if(len(newpoints) == totalPoints-1):
        newpoints.append(points[-1])
    return newpoints
```

modify the data so that the *x, y, z* axis are next to each other in the 2D array which can be represented in Figure 7.1.

A good example is: suppose we use a 2500 millisecond time window with a 500-millisecond overlap. This produced a total of at most 250 data-points per window of the *x, y, z* axis in the data. In order to reshape it to a 24x24 2d dimension, we had to manipulate the data. We first converted the 24x24 into a single dimension value producing 576 total data-sets. We then took the 576 and divided it by 3 to determine the number of points needed per axis, this resulted in 192 data points. With the sliding window producing 250 data points, we used resampling to convert it to 192. After that we reshaped the data to fit a 2d array where the *x, y, z* point location moves sequentially to the left for each row, this can be visually shown in Figure 7.1.

7.3 Convolutional Neural Network

The development of a neural network requires to have at minimum two layers an input layer and output layer. For some neural networks they may also contain hidden layers. The purpose of the hidden layers is to manipulate the input data or add weighted variation when it connects to the output.

For the convolutional neural network to connect to the output layer there is a need to connect it to a single dimensional layer. This is performed through flattening which reduces the 2-dimensional convolutional output layer to a single dimensional layer. Visually shown in Figure 7.2, the flattened layer connects to a fully connected layer which is similar to a perceptron layer. The perceptron layer uses weighted values to produce an output connection to a set of neurons equal to the classifications shown in Figure 7.2.

For our system, we used the convolutional neural network which requires a 2-dimensional input as the first layer. The convolutional filtering dimensions used is 5x5 with a 1 by 1 stride extracted from a 24x24 dimensional input. Figure 7.2 and Figure 7.3a show that we

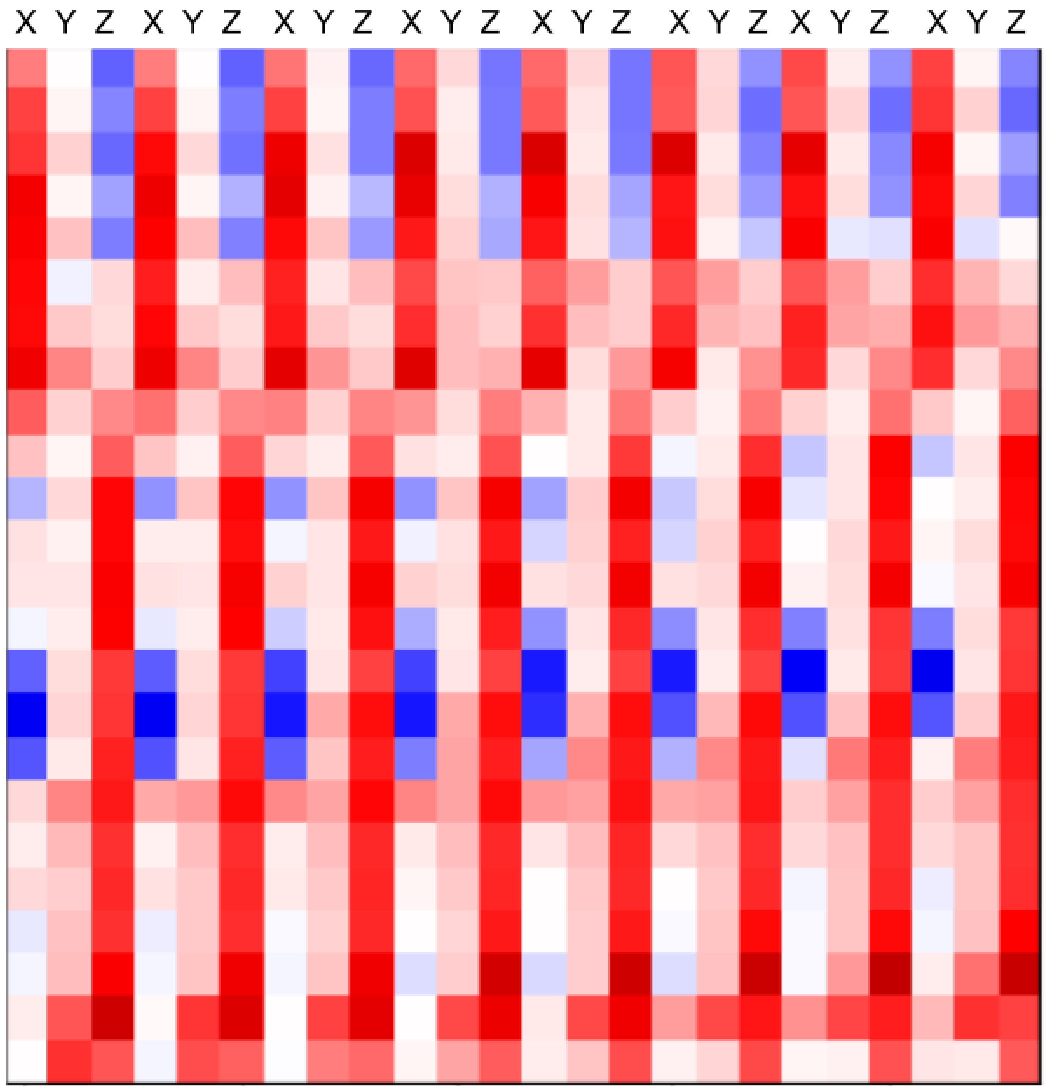


Figure 7.1: Reshape processing of accelerometer data

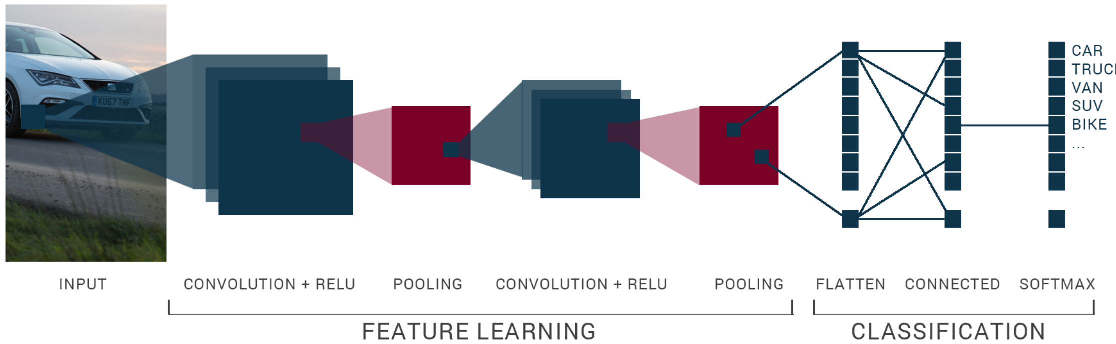
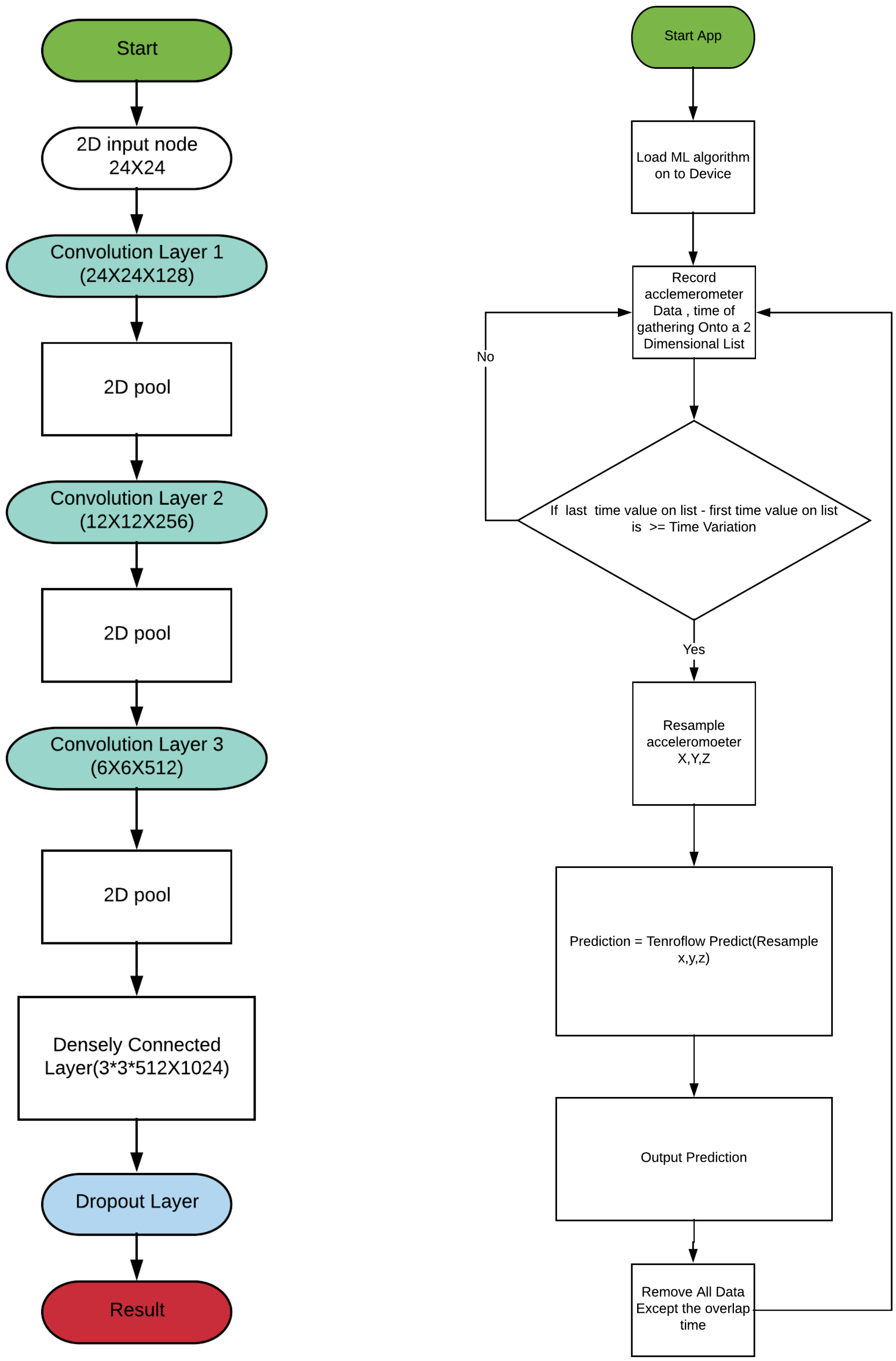


Figure 7.2: Convolutional Layer Deep Learning

connect the input layer to two more convolutional layers. This reduces the final output of the last layer to 6X6 dimensions. The next steps were to convert the 2-dimension convolutional layer into a single dimension. This step is performed during the flattening layer which makes the 6x6 to a 36 neuron layer. This layer is connected to a 1024 neuron dense connected layer which outputs to six neurons where each neuron represents the six different swim stroke styles.

7.4 Testing

To validate our developed neural network, we compared it to a generic multi-layer perceptron. The purpose is that the perceptron uses dense connected layers, which is common in most neural networks. For the perceptron neural network, we used Weka, as it was also used on the feature set as well, and offered some consistency across our studies We computed the F-measure from Weka's perceptron and compared it to our built neural network.



(a) Deep Learning Neural Network Layout (b) Application Recognition Tensorflow

Figure 7.3: Neural network flow charts

7.5 Deep Learning Neural Network Android Application

With the neural network developed we integrated it with the android application to be used in user study 2. The method we used to analyze and detect activities in real time is presented in Figure 7.3b. The library we used to develop our neural network is with TensorFlow, a deep learning system developed by Google. The reason is that Tensorflow has a python and java model integration. This allows us to train and build the model on the computer and integrate with the mobile system after completion. For the input into the neural network, we used the similar sampling method as shown in the pseudocode in Algorithm 1.

8. EVALUATION OF MAJOR SWIM STROKES RECOGNITION

From the collected data, we evaluated the recognition accuracy of major swim styles (freestyle, backstroke, breaststroke, butterfly, treading water, sidestroke). We performed two users studies with fifteen participants in the first user study and five participants in the second user study. In this chapter, we discuss the processing of the data and the results of both user studies on the machine learning algorithms. Within the chapter the ✓is traditional features, ✓novel features, ✓combination of both features.

8.1 Recognition Features Quantification

The filtered features provided insight in personal swim style and the comparison of feature groupings. We examined traditional features and novel features by comparing the features sets to each other and the combination of them both. In Figure 8.1 it shows that both traditional and novel features are present when combined. We deducted that the features displayed in the groups and the combination of groups are ruled as the dominant features. An example is the Average-z feature presented between backstroke and all other swim strokes. Figure 8.1 visualized that freestyle, backstroke, breaststroke, and sidestroke use the Average-z feature when filtered.

	Traditional			Novel			Combined		
	x	**y**	**z**				**x**	**y**	**z**
Average	✓		✓						✓
Standard Deviation	✓	✓	✓				✓	✓	✓
Root Mean Square	✓	✓	✓				✓		
Peak Count								✓	
Valley Count	✓								
Skewness							✓		✓
Kuritosis								✓	
Cross Correlation							✓	✓	✓
Entropy	✓	✓	✓				✓	✓	
				x	**y**	**z**			
Max Peak Angle				✓					
Min Peak Angle					✓		✓		
Average Peak Angle						✓			✓
Max Valley Angle				✓				✓	
Min Valley Angle				✓					
Average Valley Angle					✓	✓			
Axis Angle Average				✓		✓	✓		
Cross Correlation				✓	✓	✓	✓	✓	✓
Zero Crossing				✓	✓		✓		
DC Component				✓	✓	✓	✓	✓	✓
Power Spectral Density				✓	✓	✓			

Table 8.1: Feature Selection All Strokes Types Main Swim Strokes

Stroke Type	Freestyle			Backstroke			BreastStroke			Butterfly			Sidestroke			Treadingwater		
Axis	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
Average			✓✓			✓✓			✓✓				✓✓		✓✓		✓✓	
Standard Deviation	✓✓		✓	✓				✓✓			✓✓	✓✓						
Root Mean Square	✓	✓					✓		✓			✓	✓✓			✓✓	✓✓	
Peak Count			✓✓					✓		✓✓					✓			
Valley Count			✓			✓		✓	✓	✓✓					✓	✓		
Skewness						✓			✓									
Kuritosis				✓									✓					
Correlation Coefficeint																		
Entropy									✓	✓✓	✓		✓✓	✓	✓✓		✓✓	

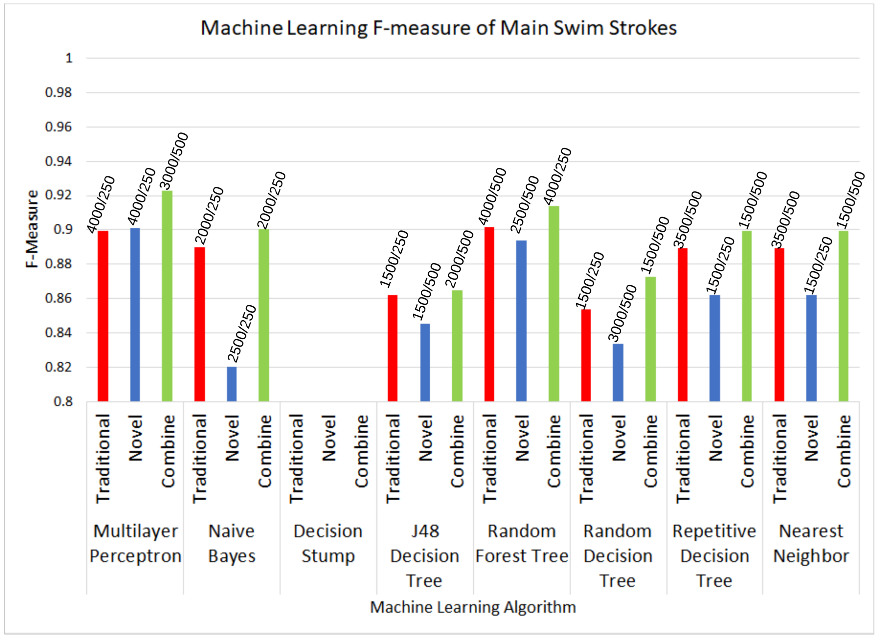
Stroke Type	Freestyle			Backstroke			BreastStroke			Butterfly			Sidestroke			Treadingwater		
Axis	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
Peak Max Angle				✓			✓✓		✓									✓
Peak Min Angle											✓	✓						
Peak Average Angle									✓	✓✓						✓✓		
Valley Max Angle	✓						✓	✓				✓				✓✓	✓	✓
Valley Min Angle																		
Valley Average Angle	✓					✓				✓✓	✓	✓✓					✓	
Total Angle						✓✓		✓✓	✓				✓		✓	✓✓	✓	
Cross Correlation	✓	✓			✓	✓✓	✓✓	✓✓	✓	✓	✓	✓						
Zero Crossing					✓✓		✓		✓	✓✓	✓		✓✓		✓			
DC Component	✓	✓	✓	✓			✓✓		✓✓			✓				✓	✓	
Power Spectral						✓✓			✓	✓			✓✓				✓	

Table 8.2: Features Selection Individual Stroke Types Main Swim Strokes

8.2 Evaluation Machine Learning Algorithms

Activity recognition depends on a machine learning algorithm that provides the best F-measure to the time window that will obtain the optimal results. The way we achieved that is we compared several machine learning algorithms as presented in Chapter 6 with multiple time windows. Figure 8.1 shows that the most optimal feature set represents the combination of traditional and novel features. With the multilayer perceptron providing .922 F-measure at a window of 3000ms with 500 millisecond overlap. Another factor that is noticed is that traditional has .90 value which is a 2% difference to the combination recognition algorithm. Though the time window is larger to 4000 milliseconds, the features implemented were half as much as the combination between both as shown in Figure 8.1.

Figure 8.1: Machine learning algorithms evaluation of subset features for major swimming strokes



Multilayer Perceptron			Naive Bayes			J48 Decision Tree			Random Forest Tree		
Traditional	0.8996	4000 / 250	Traditional	0.89	2000 / 500	Traditional	0.8619	1500 / 250	Traditional	0.9014	4000 / 500
Novel	0.9011	4000 / 250	Novel	0.8204	2500 / 250	Novel	0.8452	1500 / 500	Novel	0.8939	2500 / 500
Combine	0.9226	3000 / 500	Combine	0.9006	2000 / 250	Combine	0.865	2000 / 500	Combine	0.9138	4000 / 250
Random Decision Tree			Repetitive Decision Tree			Nearest Neighbor			Decision Stump		
Traditional	0.8535	1500 / 250	Traditional	0.8892	3500 / 500	Traditional	0.8892	3500 / 500	Traditional	0.3113	4000 / 500
Novel	0.8335	3000 / 500	Novel	0.8622	1500 / 250	Novel	0.8622	1500 / 250	Novel	0.3113	4000 / 500
Combine	0.8726	1500 / 500	Combine	0.8997	1500 / 500	Combine	0.8997	1500 / 500	Combine	0.4243	2000 / 250

8.3 Independent Validation

With the independent validation providing high F-measure values we needed to compare these values in a naturalistic setting. The way we address this is with the use of User Study 2's data as shown in Section 6.3. The results show that novel features are capable of detecting breaststroke better than butterfly stroke compared to the traditional features. When combined, the breast stroke stays accurate though butterfly stroke accuracy decreases in value. The filtering of features shows that there are two traditional features missing like entropy and Root Mean Square in the z-axis. When distinguishing between breast and butterfly strokes the y and z-axis represents the most dominant features as shown in Figure 4.4

Table 8.3: Traditional features with 4000ms time window with 500ms overlap using random forest

Freestyle	Backstroke	Breaststroke	Butterfly	Treading water	Sidestroke	Total / Percent
144/1	0/0	0/0	0/0	0/0	0/0	Freestyle
0/0	60/1	0/0	0/0	0/0	0/0	Backstroke
0/0	0/0	87/1	0/0	0/0	0/0	Breaststroke
0/0	0/0	13/.24	42/.76	0/0	0/0	Butterfly
0/0	0/0	0/0	0/0	92/1	0/0	Treading water
0/0	0/0	0/0	0/0	5/.1	48/.9	Sidestroke

Table 8.4: Novel features with 4000ms time window with 250ms overlap using multilayer perceptron

Freestyle	Backstroke	Breaststroke	Butterfly	Treading water	Sidestroke	Total / Percent
129/.97	1/.01	3/.02	0/0	0/0	0/0	Freestyle
0/0	55/1	0/0	0/0	0/0	0/0	Backstroke
0/0	0/0	77/.95	4/.05	0/0	0/0	Breaststroke
0/0	0/0	20/.41	29/.59	0/0	0/0	Butterfly
0/0	0/0	0/0	0/0	83/.98	2/.02	Treading water
0/0	0/0	1/.02	0/0	4/.08	45/.9	Sidestroke

Table 8.5: Combined features with 3000ms time window with 500ms overlap using multilayer perceptron

Freestyle	Backstroke	Breaststroke	Butterfly	Treading water	Sidestroke	Total / Percent
172/1	0/0	0/0	0/0	0/0	0/0	Freestyle
0/0	69/1	0/0	0/0	0/0	0/0	Backstroke
0/0	0/0	103/1	0/0	0/0	0/0	Breaststroke
0/0	0/0	23/.37	38/.63	0/0	0/0	Butterfly
0/0	0/0	0/0	0/0	107/1	0/0	Treading water
0/0	0/0	0/0	0/0	6/.09	59/.91	Sidestroke

9. EVALUATION OF FLIP TURNS RECOGNITION

From the collected data in the user studies, we acquired and labelled flip turns. We evaluated the recognition of flip turns swim styles. The flip turns have been separated into two categories backstroke/freestyle flip turn and breaststroke/butterfly flip turns. In this chapter, we discuss the processing of the data and the results of the data evaluated on the machine learning algorithms. Within the chapter the ✓ is traditional features, ✓ novel features, ✓ combination of both features.

9.1 Recognition Features Quantification

Compared to Section 8.1 there were two flip turns that the features set will be on. When visually looking at the flip turns as shown in Figure 4.5 there are considerable differences in the z-axis. Figure 9.1 verifies this with the filters extracting features that are in the z-axis. Feature Mean-z is presented in both combinations of features and in the traditional features group. Though when evaluating the novel features there is no inclusion of novel features in the combination of both feature sets.

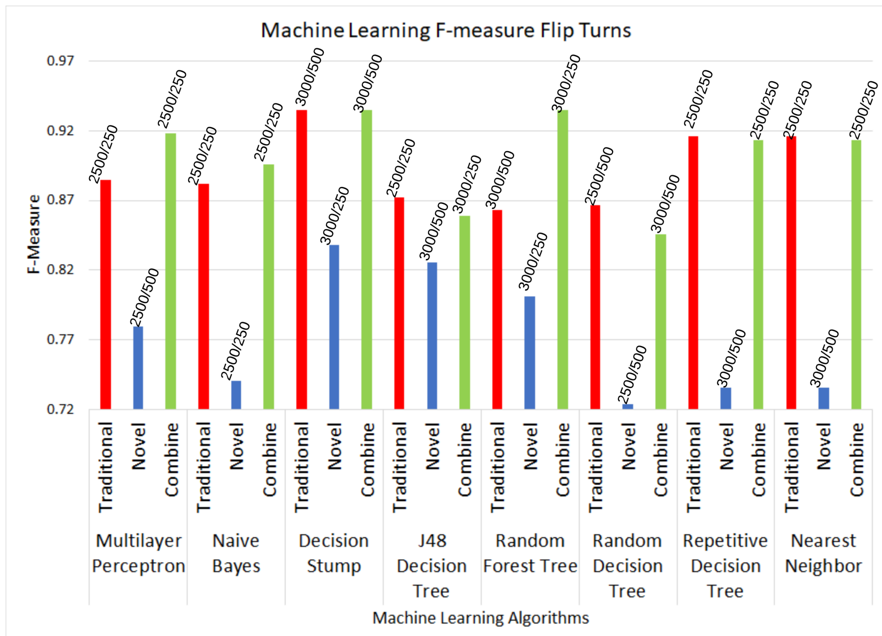
9.2 Evaluation Machine Learning Algorithms

With the reduction in the amount of features in Section 9.1 there is an expected rise in naive bayes, decision stump, and nearest neighbor recognition accuracy. We evaluated all features set with Figure 9.1 displaying that decision stump provides the highest F-measure of .9347222 in both traditional and combination of features. The decision stumps uses a single layer with a single feature as its input. In the system the decision stump used the Average Z axis which shows that both traditional and combination providing similar values. This shows that currently the traditional features set providing high recognition capabilities.

Table 9.1: Subset selection on features related to flip turns

	Traditional			Novel			Combined		
	x	**y**	**z**				**x**	**y**	**z**
Average			✓					✓	✓
Standard Deviation			✓						✓
Root Mean Square							✓	✓	
Peak Count									
Valley Count									
Skewness									✓
Kuritosis									
Cross Correlation									
Entropy									
				x	**y**	**z**			
Max Peak Angle									
Min Peak Angle									
Average Peak Angle									
Max Valley Angle									
Min Valley Angle									
Average Valley Angle									
Axis Angle Average									
Cross Correlation					✓				
Zero Crossing									
DC Component									
Power Spectral Density									

Figure 9.1: Machine learning algorithms evaluation of subset features for swimming flip turns



Multilayer Perceptron			Naive Bayes			J48 Decision Tree			Random Forest Tree		
Traditional	0.8848485	2500 / 250	Traditional	0.8815851	2500 / 250	Traditional	0.872028	2500 / 250	Traditional	0.8632479	3000 / 500
Novel	0.7794949	2500 / 500	Novel	0.740404	2500 / 250	Novel	0.8252137	3000 / 500	Novel	0.8010989	3000 / 250
Combine	0.9181818	2500 / 250	Combine	0.8959998	2500 / 500	Combine	0.8592262	3000 / 250	Combine	0.9347222	3000 / 500
Random Decision Tree			Repetitive Decision Tree			Nearest Neighbor			Decision Stump		
Traditional	0.8668908	2500 / 500	Traditional	0.9157731	2500 / 250	Traditional	0.9157731	2500 / 250	Traditional	0.9347222	3000 / 500
Novel	0.7238966	2500 / 500	Novel	0.7354701	3000 / 500	Novel	0.7354701	3000 / 500	Novel	0.8380342	3000 / 500
Combine	0.8458333	3000 / 500	Combine	0.9135522	2500 / 250	Combine	0.9135522	2500 / 250	Combine	0.9347222	3000 / 500

9.3 Independent Validation

We examine the flip turns in a naturalistic setting by handling the data from User Study 2. In the evaluation, the features show that traditional feature set provides virtually 50% accuracy of freestyle/backstroke flip turns. While the majority classifier is classifying at 47%. Though when examining features from novel classifier that 95% of breaststroke/butterfly flip turns can be classified which is .1% higher then the traditional feature sets with a similar count in features used. As shown in Figure 9.1 that in the *y*-axis was the only feature filtered and the decision stump classified highest on average on traditional features. Which means both used a single feature set in classification.

Table 9.2: Traditional features with 3000ms time window with 500ms overlap using decision stump

Total / Percent	Freestyle/Backstroke Flip	Breaststroke/Butterfly Flip
Freestyle/Backstroke Flip	35/.57	26/.43
Breaststroke/Butterfly Flip	4/.13	26/.87

Table 9.3: Novel features with 3000ms time window with 250ms overlap using decision stump

Total / Percent	Freestyle/Backstroke Flip	Breaststroke/Butterfly Flip
Freestyle/Backstroke Flip	61/.77	18/.23
Breaststroke/Butterfly Flip	2/.05	38/.95

Table 9.4: Combined features with 3000ms time window with 500ms overlap using decision stump

Total / Percent	Freestyle/Backstroke Flip	Freestyle/Backstroke Flip
Freestyle/Backstroke Flip	35/.57	26/.43
Breaststroke/Butterfly Flip	4/.13	26/.87

10. EVALUATION OF MAJOR SWIM STROKES AND FLIP TURNS RECOGNITION

From previous chapters we compared major swim styles (backstroke, breaststroke, butterfly, freestyle, sidestroke, and treading water) and flip turns (backstroke/freestyle, breaststroke/butterfly). With the results of the chapters we want to compare if possible the combination of both flip turns and main swim styles. This is done through using smaller time window sizes of 1500 to 2500. The reason for this is that flip turns actions are smaller while main swim styles have several iterations based on when they breath and motion. From the collected data we evaluated the features, machine learning recognition, and validations of both flip turns and major swim styles. In this chapter we discuss the data processing and the results of the data evaluated on the machine learning algorithms.

10.1 Recognition Features Quantification

Section 8.1 and Section 9.1 showed that average-z and average-x are the most optimal features when distinguishing between flip turns and backstroke. We examined both flip turns and swim styles and found an decrease in feature count to 23 features when combined of main swim strokes was 25. The most desired features are Entropy, Correlation Coefficient, and DC Component. DC Component and Entropy are frequency-based features with novel and traditional both presented in the combination of feature sets. When examining individual swim styles both flip turns when filtered did not select the features in the x-axis.

Filtering features distinguish and separates the features that have a causation or a correlation to understanding the motion as of each activity. The filtering was used on traditional, novel, and the combination of both features when used on the data set of User Study 1. Table 10.1 shows that when separated specific features were presented to have more of an effect in the recognition of main stroke activities.

Table 10.1: Subset selection for major swim strokes and flip turns

	Traditional			Novel			Combined		
	x	**y**	**z**				**x**	**y**	**z**
Average	✓		✓				✓		✓
Standard Deviation	✓	✓	✓				✓	✓	✓
Root Mean Square	✓	✓	✓				✓		
Peak Count	✓	✓	✓					✓	✓
Valley Count	✓	✓	✓						
Skewness									✓
Kuritosis							✓		
Cross Correlation									
Entropy	✓	✓	✓				✓	✓	
				x	**y**	**z**			
Max Peak Angle				✓	✓	✓	✓		
Min Peak Angle									
Average Peak Angle									✓
Max Valley Angle				✓	✓	✓			✓
Min Valley Angle									
Average Valley Angle									
Axis Angle Average						✓	✓		
Cross Correlation				✓	✓	✓	✓	✓	✓
Zero Crossing				✓	✓	✓	✓		
DC Component				✓	✓	✓	✓	✓	✓
Power Spectral Density				✓	✓	✓			

Table 10.2: Features Selection Individual Stroke Types Main Swim Strokes

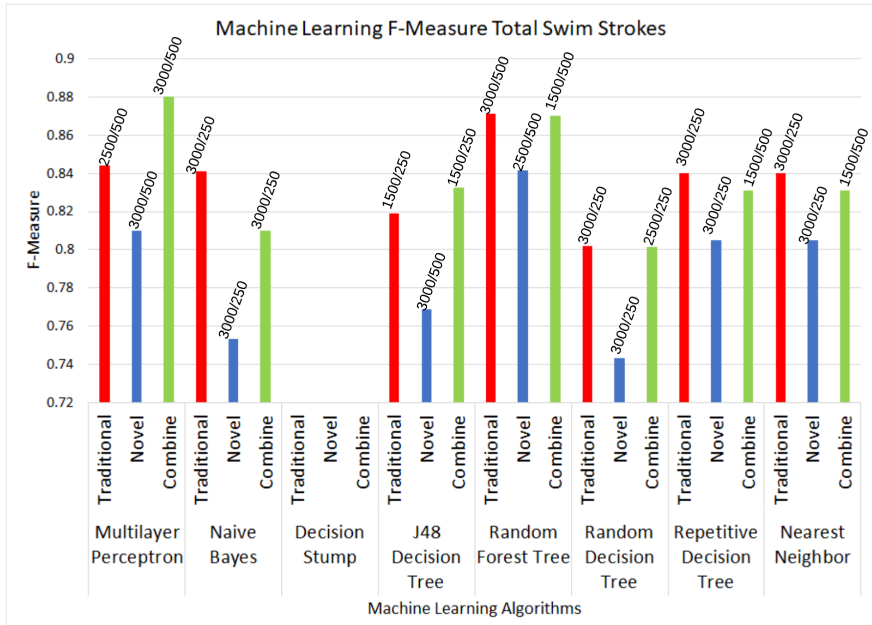
Stroke Type	Freestyle			Backstroke			BreastStroke			Butterfly			Sidestroke			Treadingwater			Backstroke/Freestyle Flipturn			Breaststroke/Butterfly Flipturn		
Axis	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
Average			✓✓			✓✓			✓✓			✓	✓✓				✓✓	✓						✓✓
Standard Deviation	✓✓			✓		✓		✓✓			✓✓	✓✓				✓✓			✓	✓	✓✓	✓	✓✓	✓✓
Root Mean Square		✓✓					✓✓		✓	✓		✓✓	✓✓		✓✓	✓✓	✓	✓						
Peak Count	✓		✓✓		✓			✓	✓	✓✓	✓	✓	✓	✓					✓✓	✓	✓	✓	✓	✓✓
Valley Count	✓	✓	✓✓		✓					✓	✓					✓			✓✓	✓		✓✓	✓	✓✓
Skewness						✓														✓				
Kuritosis				✓									✓								✓			
Correlation Coefficeint																								
Entropy									✓✓	✓✓	✓✓		✓		✓		✓✓			✓	✓			✓

Stroke Type	Freestyle			Backstroke			BreastStroke			Butterfly			Sidestroke			Treadingwater			Backstroke/Freestyle Flipturn			Breaststroke/Butterfly Flipturn		
Axis	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
Peak Max Angle	✓			✓			✓✓									✓✓								
Peak Min Angle																								
Peak Average Angle									✓		✓	✓						✓						
Valley Max Angle	✓✓	✓✓	✓					✓			✓	✓				✓	✓	✓✓	✓				✓	
Valley Min Angle																								
Valley Average Angle						✓	✓			✓													✓	
Average Combined Axis				✓		✓	✓	✓	✓							✓✓								
Cross Correlation	✓	✓		✓	✓		✓✓	✓✓	✓	✓	✓	✓		✓		✓				✓✓	✓		✓	
Zero Crossing	✓				✓✓				✓	✓			✓✓		✓		✓				✓			
DC Component	✓✓		✓			✓	✓✓		✓			✓			✓		✓							
Power Spectral		✓	✓				✓		✓				✓✓				✓✓				✓			

10.2 Evaluation Machine Learning Algorithms

For machine learning algorithms we examined on a smaller time varying window of 1500 to 3000 milliseconds. This is due to the single flip turn is a single iterations and lasts less than 3000 milliseconds when analyzed. We examined the multiple machine learning algorithms and found .88 F-measure is the highest with combination feature set. traditional is .87 F-measure from a random forest. When examining the times both traditional and combination used 3000 millisecond time window. Second highest is random forest with 1500 millisecond time window using combination of both feature sets.

Figure 10.1: Machine learning algorithms evaluation of subset features for major swimming strokes and flip turns



(a) Machine Learning Algorithms Evaluation

Multilayer Perceptron			Naive Bayes			J48 Decision Tree			Random Forest Tree		
Traditional	0.8442944	2500 / 500	Traditional	0.841348	3000 / 250	Traditional	0.8191712	1500 / 250	Traditional	0.871019	3000 / 500
Novel	0.809792	3000 / 500	Novel	0.7532429	3000 / 250	Novel	0.7690283	3000 / 500	Novel	0.841635	2500 / 500
Combine	0.8803225	3000 / 500	Combine	0.8101258	3000 / 250	Combine	0.8325177	1500 / 250	Combine	0.8700408	1500 / 500
Random Decision Tree			Repetitive Decision Tree			Nearest Neighbor			Decision Stump		
Traditional	0.8019997	3000 / 250	Traditional	0.840087	3000 / 250	Traditional	0.840087	3000 / 250	Traditional	0.2747173	3000 / 500
Novel	0.7430475	3000 / 250	Novel	0.8050006	3000 / 250	Novel	0.8050006	3000 / 250	Novel	0.2649611	2000 / 250
Combine	0.8012022	2500 / 250	Combine	0.831065	1500 / 500	Combine	0.831065	1500 / 500	Combine	0.2478193	1500 / 250

10.3 Independent Validation

When it comes to a naturalistic setting there was a need to examine all swim styles based on the features set and machine learning algorithms. We evaluated traditional, novel, and combined features sets with the optimal machine learning algorithms. It showed that all groups can classify freestyle, backstroke, breaststroke, treading water at 100% accuracy. Though both traditional and novel have issues in classification of flip turns. The percent accuracy decrease by almost 30% for detecting breaststroke/butterfly flip turns. Though when evaluating the combination there was a slight increase by .01 when evaluating flip urns and sidestroke. Which currently shows that the combination of traditional and novel features has a slight increase.

Table 10.3: Traditional features with 3000ms time window with 500ms overlap using random forest

Total Percent	Freestyle	Backstroke	Breaststroke	Butterfly	Treadingwater	Sidestroke	Freestyle/Backstroke Flip	Breaststroke/Butterfly Flip
Freestyle	208/1	0/0	0/0	0/0	0/0	0/0	0/0	0/0
Backstroke	0/0	85/1	0/0	0/0	0/0	0/0	0/0	0/0
Breaststroke	0/0	0/0	125/1	0/0	0/0	0/0	0/0	0/0
Butterfly	0/0	0/0	27/.35	50/.65	0/0	0/0	0/0	0/0
Treadingwater	0/0	0/0	0/0	0/0	126/1	0/0	0/0	0/0
Sidestroke	0/0	0/0	0/0	0/0	7/.09	69/.90	0/0	0/0
Freestyle/Backstroke Flip	5/.08	4/.06	1/.02	0/0	0/0	1/.02	45/.74	5/.08
Freestyle/Backstroke Flip	2/.06	0/0	3/.1	4/.13	0/0	0/0	3/.1	18/.61

Table 10.4: Novel features with 2500ms time window with 500ms overlap using random forest

Total Percent	Freestyle	Backstroke	Breaststroke	Butterfly	Treadingwater	Sidestroke	Freestyle/Backstroke Flip	Breaststroke/Butterfly Flip
Freestyle	268/1	0	0	0	0	0	0	0
Backstroke	0	109/1	0	0	0	0	0	0
Breaststroke	0	0	155/1	0	0	0	0	0
Butterfly	0	0	50/.52	46/.48	0	0	0	0
Treadingwater	0	0	0	0	160/1	0	0	0
Sidestroke	0	0	0	0	8/.08	91/.92	0	0
Freestyle/Backstroke Flip	20/.08	7/.06	1/.02	0	0	2/.02	50/.74	1/.08
Freestyle/Backstroke Flip	9/.22	0	12/.3	1/.02	0	2/.05	7/.17	10/.24

Table 10.5: Combined features with 3000ms time window with 500ms overlap using multilayer perceptron

Total Percent	Freestyle	Backstroke	Breaststroke	Butterfly	Treadingwater	Sidestroke	Freestyle/Backstroke Flip	Breaststroke/Butterfly Flip
Freestyle	208/1	0	0	0	0	0	0	0
Backstroke	0	85/1	0	0	0	0	0	0
Breaststroke	0	0	85/1	0	0	0	0	0
Butterfly	0	0	18/.23	59/.77	0	0	0	0
Treadingwater	0	0	0	0	127/1	0	0	0
Sidestroke	0	0	0	0	7/.09	69/.91	0	0
Freestyle/Backstroke Flip	5/.08	0	0	0	0	0	52/.85	4/.06
Freestyle/Backstroke Flip	4/.13	0	3/.1	1/.03	0	0	2/.07	20/.67

11. RECOGNITION EVALUATION DEEP LEARNING SYSTEM

A deep learning neural network is frequently utilized as a black box. This means that features can't be extracted easily due to the network reliance on weighted connecting values. We developed a system that uses Tensorflow to implement convolutional layers and is evaluated on 70/30 data split among the data of User Study 1. We stored the model and integrated it into an Android mobile application. We developed the application to recognize swim styles in real time which we used in a real-world setting as presented in Section 7.2. In this chapter, we will present the neural network developed and results from a real-world setting.

11.1 Development And Training

When considering the development of a neural network we need to consider the number of layers and input/output nodes. We used a python library Tensorflow to develop the neural network. This allowed us to construct a deep learning system that will be able to be integrated into an Android device. For the development, we built 3 connecting convolutional layers and a dense layer as shown in Figure 7.3a. Section 7.2 shows how we modified the data to be 512 nodes visually indicated in Figure 7.1. Allowing for reduction as the data passes through the convolutional inputs as shown in Figure 11.2a.

For training the data we had to examine the number of epochs which is a group of batches. Batches are groups that hold a number of data sets. For training, we use 10 epochs which contain 50 batches with each batch holding 50 datasets. In total 5000 datasets are examined per training session. After 10 epochs we tested the neural network on 30% of the data to get the deep learning F-measure and ability to recognize swimming activity. The results per every 10 epochs in training of the neural network can be presented in Figure 11.1. Altogether the system was able to acquire 99% F-measure with a small number of epochs required.

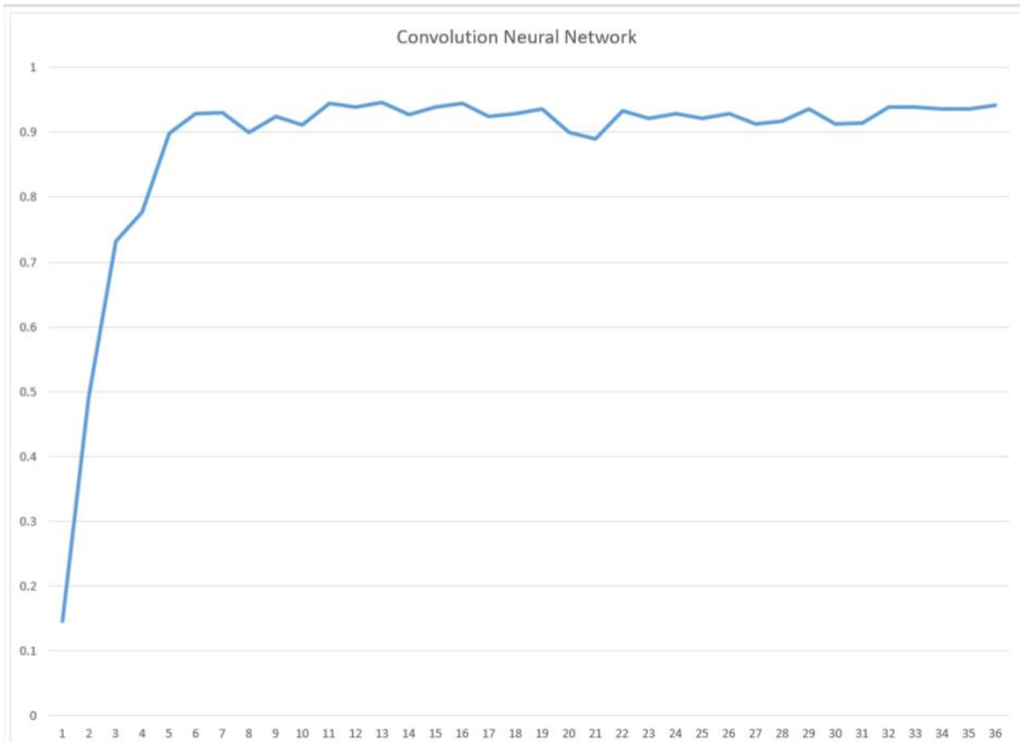
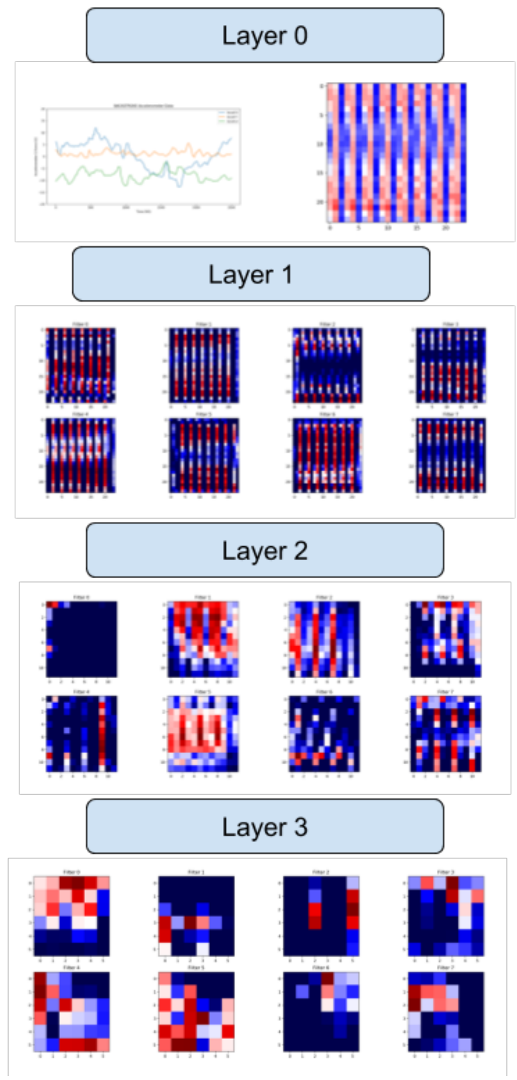


Figure 11.1: Visualization graph of the F-measure for training Deep Learning Neural Network

11.2 Main Strokes Swim Styles

For testing in a naturalistic setting, we developed an application that was integrated with the model developed by Tensorflow. The application developed used the same UI from User Study 1 as shown in Figure 3.1. We evaluated the system on a lone swimmer and were capable of getting 98% accuracy in real time. With side stroke having 100% classification on each side. All strokes were classified correctly except butterfly with 85% accuracy. We hope to develop a more effective system that will allow for a reduction in the incorrect classification of the butterfly swim style.

(a) Visualization graph of the filters for layers of the convolutional neural network



(b) Deep learning neural network real time evaluation

Total / Percent	Freestyle	Backstroke	Breaststroke	Butterfly	Treading water	Sidestroke
Freestyle	20/1	0	0	0	0	0
Backstroke	0	20/1	0	0	0	0
Breaststroke	0	0	20/1	0	0	0
Butterfly	0	0	3/.15	17/.85	0	0
Treading water	0	0	0	0	20/1	0
Sidestroke	0	0	0	0	0	40/1

12. DISCUSSION

The results have provided some information on understanding the multiple algorithms and time windows on a set of filtered features. We obtain a sufficient amount of data analyzed, and we plan to understand all the swimming styles that is expected for survival swimming. In this chapter we provide insight in the features, machine learning algorithms, and deep learning algorithm.

12.1 Features Insight

We grouped the feature sets into two categories (traditional and novel) and compared them to each other and the combination of both. We found that traditional features provided better accuracy in the butterfly stroke, with novel features providing higher accuracy with breaststroke, as shown in Table 8.3. We believe this is because the peaks and valleys features help to distinguish the difference between the breast and butterfly stroke. One issue is the noise that the accelerometer provides causing multiple detections when searching for peaks. The way we prepare to address that situation is by implementing a re-sampling method to reduce the points that cause noise and increase smoothing.

For classifications, we found that the combination of features sets always provided the highest f-measure. Though the combination of features increased the number of features used and the overall output. Section 10.2 has provided the combined features had only .01 of a difference in F-measure when using all swim styles. Results of the features are due to the time window being really small causing the peak detection to only detect 1 peak causing issues when finding max and min. We plan to rectify this is by evaluating only on the peaks and increasing the time window to a different time interval. This is an issue when it comes to flip turns. Figure 4.4 and Figure 4.5 shows that the peaks are similar in the y and z direction, but in the x direction there is a significant shift in value sets. This is because the x-axis focuses on the motions of the person's body to move side to side. While breaststroke and butterfly focus on up and down motions, unlike backstroke or freestyle.

12.2 Algorithm Insight

With the multiple machine learning algorithms used we compare several time windows. This was achieved through independent validation and naturalistic validation. When comparing the algorithms several stuck out, for example, Random Forest and multilayer perceptron usually had the highest yield in classification. This is due to both classifications rely on large feature sets which were provided even after filtering. Though for the smaller feature sets as shown in Section 9.2 it shows independent validation had decision stump providing the highest f-measure. That is because there is a considerable distinction when it comes to both flip turns. When combined all swim styles we could find the smaller time windows could not detect as accurately. Causing the data in the time windows to visually look similar. Like sidestroke looking similar to the concluding parts of the flip turns as shown in Figure 4.4 and Figure 4.5. Overall understanding the modified algorithms that are best for swim styles are multilayer perceptron and random forest decision tree.

12.3 Deep Learning Insight

A deep learning model was designed and implemented in a real-world setting. The system had 100% accuracy in detecting between all major swim styles except butterfly in real time. Butterfly had 85% which is an 8% higher among major swim styles recognition and total swim styles recognition. The deep learning recognition system is fast and efficient with the ability to recognize in real time. Though if we were to expand on specifics like peak detection, the deep learning would have a flaw in its ability to recognize peaks locations. The deep learning system maintains better efficiency in recognizing all major swim styles with the highest f-measure. In real time the system was capable to recognize swim strokes fast and at 97% accuracy.

13. FUTURE WORK

With the activity recognition of our system and the current statistics on swimmers, there are several other areas of research this can expand on. With the current results, we believe the recognition algorithm can serve as a form of restructuring the education, sports, health, and safety fields.

13.1 Peak Analysis

For our system, we analyzed we discovered the peak and valley detection method can be optimized to better fit multiple swimming proficiency levels. When analyzing the data the x-axis for breaststroke/butterfly flip turns can be used to distinguish between is designated swim strokes. With the ability to resampling the data we can eliminate noise as well increase the ability to distinguish all swim styles.

With understanding and finding exact peaks we will, in addition, implement better complex algorithms that are related to data between peak sets. Swimmers maintain varying distances between peaks because swimmers possess various levels in their ability to hold their breath. If we analyze the data between peaks, we may be capable to recognize the swimmer's proficiency level. With the added bonus of recognizing the swimmers breathing ability and physical stress, the workout is providing.

13.2 Education and Sports

With the user studies, we examined there were distinctions between professional swimming swim styles and beginner/inexperienced swim styles. This has presented the possibilities as our system with more data be able to provide feedback to swimmer while they perform swimming style. This will allow for real-time recognition and feedback to correct swimmers stroke style. Recognition can additionally be implemented as a testing system to evaluate peoples capability to swim. With an ability to provide feedback on recommendations to swim school or certain recreational activities that best fit their swimming expertise.

13.3 Navigation

Navigation can be implemented for safety, sports, and health. By utilizing haptic feedback our system could navigate distressed people in the water to a safe location. We can additionally use the navigation in professional contexts for triathletes, long distant swimmers, and open water competitive swimmers. The reason for haptic feedback is that it has provided significant results in navigation without an increase in impairment from users reactions. Haptic solutions for intuitive navigation feedback and obstacle navigation for paratroopers [93–96], motorcyclists [97], first responders [98], construction workers [99], dementia patients [100], pedestrians [101, 102], the visually impaired [103–105], and even children and adult gamers [106, 107]. The data show that haptic navigation sustains a reduced cognitive load compared to both audio and visual navigation. This will allow us to have the ability to process an immense amount of haptic information, especially in emergency scenarios, because it is an underused sensory channel [108–110].

14. CONCLUSION

The purpose of this research was to understand swim styles related to proficiency examinations and survival swimming techniques. To be a proficient swimmer and acquire the skills to survive water based activity and reduce the risk of drowning, a person must be capable to learn how to breaststroke, sidestroke, freestyle, backstroke, and tread water. Current research focuses on the recognition of the competitive swim styles (backstroke, breaststroke, freestyle, butterfly).

Our research focuses on the competitive swim styles as well as other motions related to recreational swimming and survival swimming like sidestroke, treading water, and flip turns. This paper presented several methodologies in gathering data with the use of a mobile device and ways to setup the system through wearable systems. We also show current features and novel features used in recognizing these swim styles and which features will indicate the most reliable results. With these features, we present and compare multiple machine learning algorithms and found the best time window that produced the highest f-measure. These algorithms help when recognizing swim styles, flip turns, and the combination of both.

14.1 Methodology

With the advancement of technology and the ability that mobile devices are able to perform. We wanted to develop a system that will allow us to collect swimming data. Two user studies required different methods in extracting features. User Study 1 allowed an individual to interact with the mobile device to extract data. This allowed for others to develop a general system that can allow participants to provide their data. For User Study 2 it requires a proctor to record swimming styles but allowing the swimmer to maintain a constant motion. With the ability to swim for a more extended period of time. With the increase in mobile system, researchers can employ our system to advance the knowledge of swimming basted studies.

14.2 Features

For features, we compared traditional time window features to novel peak related feature sets. We discovered that peak related features produced similar f-measures when related to broader windows as shown in Chapter 8. When it came to flip turns and the combination of flip turns and major swim styles. We evaluated that when the time window was reduced it extracted much less novel feature sets but produced similar results when traditional and combined features were produced. With more work, we believe that the peak related features will allow us to recognize optimally. With the added bonus of extracting motions related to a person performing a swim stroke and breathing motion in the water.

14.3 Machine Learning Algorithm

Machine learning algorithms remain a fundamental aspect of recognizing swim styles. We analyzed several machine learning algorithms to provide the optimal algorithm to the best time window. We analyzed seven machine learning algorithms all having pros and cons based on time window and features. We discovered when it came to the major swim styles that novel and traditional had an f-measure of .91. The best sliding window obtained is 4000 milliseconds. The pros in implementing the novel features is the ability to count the peaks which exhibit the most significant correlation to the locomotion of the swimming style.

For understanding, all swim styles we compared it to major and flip turns separated. It shows the smaller time window reduces the novel features to recognize better as shown in major swim styles requires a longer time. This means there needs to be an improvement in the peak detection algorithm. To understand flip turns as a whole and utilize these peaks and compare it to peaks of the other swim styles.

14.4 Deep Learning Algorithm

For the deep learning algorithm, we wanted to analyze and determine if a neural network can provide higher accuracy and be able to recognize faster. We developed a convolutional neural network based on its capabilities in recognizing images based on pixel color. As presented in Section 4.2, we employed a similar means to convert the time windows data to a two-dimensional input as shown in Figure 7.1. This was then implemented and tested on an actual swimmer in real time. It produced the highest accuracy at 98%. The value is greater than the featured recognition system which can analyze at 90%. Currently, this system is the most optimal form in detecting swimming styles. The limit of a neural network is in the ability to detect peak values. This will require more research, and we believe there are better methods in extracting peaks and inevitably the locomotion.

REFERENCES

[1] Daniel Courteix, Philippe Obert, Anne-Marie Lecoq, Patrick Guenon, and Günter Koch. Effect of intensive swimming training on lung volumes, airway resistances and on the maximal expiratory flow-volume relationship in prepubertal girls. *European journal of applied physiology and occupational physiology*, 76(3):264–269, 1997.

[2] World Health Organization. *Global report on drowning: preventing a leading killer*. WHO Press, 2014.

[3] Center for Disease Control and Prevention. Unintentional drowning: Get the facts, apr 2016.

[4] Red Cross. Red cross launches campaign to cut drowning in half in 50 cities, May 2014.

[5] Red Cross. Red cross finds 61% of children can't swim, June 2015.

[6] Gary B Zuckerman and Edward E Conway. Drowning and near drowning: a pediatric epidemic. *Pediatric annals*, 29(6):360–364, 2000.

[7] Ruth A Brenner, Gitanjali Saluja, and Gordon S Smith. Swimming lessons, swimming ability, and the risk of drowning. *Injury control and safety promotion*, 10(4):211–215, 2003.

[8] Robert Keig Stallman, Malfrid Junge, and Turid Blixt. The teaching of swimming based on a model derived from the causes of drowning. *International Journal of Aquatic Research and Education*, 2(4):11, 2008.

[9] Carolyn Staines and Joan Ozanne-Smith. Drowning deaths between 1861 and 2000 in victoria, australia. *Bulletin of the World Health Organization*, 95(3):174, 2017.

[10] Gail H Ito. Barriers to swimming and water safety education for african americans.

International journal of aquatic research and education, 8(3):4, 2014.

[11] Michael Brooks. *Developing swimmers*. Human Kinetics, 2011.

[12] Scott M Vontroba. A cognitive component to adult swim lessons: an instruction manual. Master's thesis, California State University, Sacramento, Sacramento, CA, 8 2011.

[13] David French and Jenny Hainsworth. 'there aren't any buses and the swimming pool is always cold!': obstacles and opportunities in the provision of sport for disabled people. *Managing Leisure*, 6(1):35–49, 2001.

[14] Apple watch, jun 2018.

[15] Akin Avci, Stephan Bosch, Mihai Marin-Perianu, Raluca Marin-Perianu, and Paul Havinga. Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In *Architecture of computing systems (ARCS), 2010 23rd international conference on*, pages 1–10. VDE, 2010.

[16] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

[17] Ling Bao and Stephen S Intille. Activity recognition from user-annotated acceleration data. In *International Conference on Pervasive Computing*, pages 1–17. Springer, 2004.

[18] Oscar D Lara and Miguel A Labrador. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys and Tutorials*, 15(3):1192–1209, 2013.

[19] Tim Van Kasteren, Athanasios Noulas, Gwenn Englebienne, and Ben Kröse. Accurate activity recognition in a home setting. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 1–9. ACM, 2008.

[20] Jeffrey W Lockhart, Tony Pulickal, and Gary M Weiss. Applications of mobile

activity recognition. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 1054–1058. ACM, 2012.

[21] Keith M Diaz, David J Krupka, Melinda J Chang, James Peacock, Yao Ma, Jeff Goldsmith, Joseph E Schwartz, and Karina W Davidson. Fitbit®: An accurate and reliable device for wireless physical activity tracking. *International journal of cardiology*, 185:138–140, 2015.

[22] Fatema El-Amrawy and Mohamed Ismail Nounou. Are currently available wearable devices for activity tracking and heart rate monitoring accurate, precise, and medically beneficial? *Healthcare informatics research*, 21(4):315–320, 2015.

[23] Lina M Castano and Alison B Flatau. Smart fabric sensors and e-textile technologies: a review. *Smart Materials and Structures*, 23(5):053001, 2014.

[24] Ali K Yetisen, Hang Qu, Amir Manbachi, Haider Butt, Mehmet R Dokmeci, Juan P Hinestroza, Maksim Skorobogatiy, Ali Khademhosseini, and Seok Hyun Yun. Nanotechnology in textiles. *ACS nano*, 10(3):3042–3068, 2016.

[25] Joanna Berzowska. Electronic textiles: Wearable computers, reactive fashion, and soft computation. *Textile*, 3(1):58–75, 2005.

[26] Anne Cranny-Francis. *Technology and touch: The biopolitics of emerging technologies*. Springer, 2013.

[27] Peter Blank, Julian Hoßbach, Dominik Schuldhaus, and Bjoern M Eskofier. Sensor-based stroke detection and stroke type classification in table tennis. In *Proceedings of the 2015 ACM International Symposium on Wearable Computers*, pages 93–100. ACM, 2015.

[28] Xi Long, Bin Yin, and Ronald M Aarts. Single-accelerometer-based daily physical activity classification. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 6107–6110. IEEE, 2009.

[29] Joe Marshall. Smartphone sensing for distributed swim stroke coaching and research. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*, pages 1413–1416. ACM, 2013.

[30] Woohyeok Choi, Jeungmin Oh, Darren Edge, Joohyun Kim, and Uichin Lee. Swimtrain: exploring exergame design for group fitness swimming. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 1692–1704. ACM, 2016.

[31] Woohyeok Choi, Jeungmin Oh, Taiwoo Park, Seongjun Kang, Miri Moon, Uichin Lee, Inseok Hwang, and Junehwa Song. Mobydick: an interactive multi-swimmer exergame. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, pages 76–90. ACM, 2014.

[32] Josh Cherian. Recognition of everyday activities through wearable sensors and machine learning. Master's thesis, Texas A&M University, College Station, TX, USA, August 2017.

[33] Joey Bartley, Jonathon Forsyth, Prachi Pendse, Da Xin, Garrett Brown, Paul Hagseth, Ashish Agrawal, Daniel W. Goldberg, and Tracy Hammond. World of workout: A contextual mobile rpg to encourage long term fitness. In *Proceedings of the Second ACM SIGSPATIAL International Workshop on the Use of GIS in Public Health*, HealthGIS '13, pages 60–67, New York, NY, USA, 2013. ACM.

[34] Danielle Cummings. *Multimodal Interaction for Enhancing Team Coordination on the Battlefield*. PhD Doctoral Dissertation, Texas A&M University, College Station, TX, USA, August 2013. Advisor: Tracy Hammond.

[35] J. Elizondo, J. Cherian, D.W. Goldberg, and T.A. Hammond. Utilizing accelerometer data to recognize pacing with machine learning. In *Proceedings of Texas A&M University Undergraduate Research Summer Poster Session*, College Station, TX, USA, August 3, 2016.

[36] A. Gámez, Cherian, J. Goldberg, and T.A. Hammond. Mapping ultraviolet radiation with wearables. In *Proceedings of Texas A&M University Undergraduate Research Summer Poster Session*, College Station, TX, USA, August 3, 2016.

[37] Javier Anguas, Christopher Cortes, Omar Garcia, Aldo Haro, and Tracy Hammond. Smart cap (air quality and ultraviolet radiation monitoring). In *Proceedings of Texas A&M University Undergraduate Research Summer Poster Session*, College Station, TX, USA, August 2015.

[38] Vijay Rajanna, Patrick Vo, Jerry Barth, Matthew Mjelde, Trevor Grey, Cassandra Oduola, and Tracy Hammond. Kinohaptics: An automated, wearable, haptic assisted, physio-therapeutic system for post-surgery rehabilitation and self-care. *Journal of Medical Systems*, 40(3):1–12, 2015.

[39] Joshua Kirby, James Granger, Austin Griffin, Mateo Aviles, Johnny Yoon, Raniero Lara Garduno, and Tracy Hammond. Luminote: Computer aided music composition. In *Texas A&M University CSCE 482 Senior Capstone Project*, December 2016.

[40] Jace Miller and Tracy Hammond. Wiiolin: A virtual instrument using the wii remote. In *Proceedings of the 2010 Conference on New Interfaces for Musical Expression (NIME)*, pages 497–500, Sydney, Australia, June 15-18 2010.

[41] Akin Avci, Stephan Bosch, Mihai Marin-Perianu, Raluca Marin-Perianu, and Paul Havinga. Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In *Architecture of computing systems (ARCS), 2010 23rd international conference on*, pages 1–10. VDE, 2010.

[42] Emmanuel Munguia Tapia, Stephen S Intille, and Kent Larson. *Activity recognition in the home using simple and ubiquitous sensors*. Springer, 2004.

[43] Jonathan Lester, Tanzeem Choudhury, and Gaetano Borriello. A practical approach to recognizing physical activities. In *Pervasive Computing*, pages 1–16. Springer,

2006.

[44] F Foerster, M Smeja, and J Fahrenberg. Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring. *Computers in Human Behavior*, 15(5):571–583, 1999.

[45] Uwe Maurer, Asim Smailagic, Daniel P Siewiorek, and Michael Deisher. Activity recognition and monitoring using multiple sensors on different body positions. In *Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. International Workshop on*, pages 4–pp. IEEE, 2006.

[46] Jonathan Lester, Tanzeem Choudhury, Nicky Kern, Gaetano Borriello, and Blake Hannaford. A hybrid discriminative/generative approach for modeling human activities. In *IJCAI*, volume 5, pages 766–772. Citeseer, 2005.

[47] Juha Pärkkä, Miikka Ermes, Panu Korpipää, Jani Mäntyjärvi, Johannes Peltola, and Ilkka Korhonen. Activity classification using realistic data from wearable sensors. *Information Technology in Biomedicine, IEEE Transactions on*, 10(1):119–128, 2006.

[48] Ling Bao and Stephen Intille. Activity recognition from user-annotated acceleration data. *Pervasive computing*, pages 1–17, 2004.

[49] Vo Quang Viet, Hoang Minh Thang, and Deok-Jai Choi. Balancing precision and battery drain in activity recognition on mobile phone. In *Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on*, pages 712–713. IEEE, 2012.

[50] Susanna Pirttikangas, Kaori Fujinami, and Tatsuo Nakajima. Feature selection and activity recognition from wearable sensors. In *International Symposium on Ubiquitious Computing Systems*, pages 516–527. Springer, 2006.

[51] Shuangquan Wang, Jie Yang, Ningjiang Chen, Xin Chen, and Qinfeng Zhang. Human activity recognition with user-free accelerometers in the sensor networks.

In *Neural Networks and Brain, 2005. ICNN&B'05. International Conference on*, volume 2, pages 1212–1217. IEEE, 2005.

[52] Yago Saez, Alejandro Baldominos, and Pedro Isasi. A comparison study of classifier algorithms for cross-person physical activity recognition. *Sensors*, 17(1):66, 2016.

[53] Robert DiPietro, Colin Lea, Anand Malpani, Narges Ahmidi, S Swaroop Vedula, Gyusung I Lee, Mija R Lee, and Gregory D Hager. Recognizing surgical activities with recurrent neural networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 551–558. Springer, 2016.

[54] Earnest Paul Ijjina and Krishna Mohan Chalavadi. Human action recognition using genetic algorithms and convolutional neural networks. *Pattern recognition*, 59:199–212, 2016.

[55] Daniel Olguın Olguın and Alex Sandy Pentland. Human activity recognition: Accuracy across common locations for wearable sensors. In *Proceedings of 2006 10th IEEE International Symposium on Wearable Computers, Montreux, Switzerland*, pages 11–14. Citeseer, 2006.

[56] Hristijan Gjoreski, Mitja Lustrek, and Matjaz Gams. Accelerometer placement for posture recognition and fall detection. In *Intelligent environments (IE), 2011 7th international conference on*, pages 47–54. IEEE, 2011.

[57] Louis Atallah, Benny Lo, Rachel King, and Guang-Zhong Yang. Sensor positioning for activity recognition using wearable accelerometers. *IEEE transactions on biomedical circuits and systems*, 5(4):320–329, 2011.

[58] Andrea Mannini, Stephen S Intille, Mary Rosenberger, Angelo M Sabatini, and William Haskell. Activity recognition using a single accelerometer placed at the wrist or ankle. *Medicine and science in sports and exercise*, 45(11):2193, 2013.

[59] Alberto G Bonomi, Annelies HC Goris, Bin Yin, and Klaas R Westerterp. Detection

of type, duration, and intensity of physical activity using an accelerometer. *Medicine & Science in Sports & Exercise*, 41(9):1770–1777, 2009.

[60] Jörg Spörri, Josef Kröll, Benedikt Fasel, Kamiar Aminian, and Erich Müller. The use of body worn sensors for detecting the vibrations acting on the lower back in alpine ski racing. *Frontiers in physiology*, 8:522, 2017.

[61] Oliver Amft and Gerhard Tröster. Recognition of dietary activity events using on-body sensors. *Artificial intelligence in medicine*, 42(2):121–136, 2008.

[62] Kai Kunze and Paul Lukowicz. Sensor placement variations in wearable activity recognition. *IEEE Pervasive Computing*, 13(4):32–41, 2014.

[63] Shoya Ishimaru, Kai Kunze, Koichi Kise, Jens Weppner, Andreas Dengel, Paul Lukowicz, and Andreas Bulling. In the blink of an eye: combining head motion and eye blink frequency for activity recognition with google glass. In *Proceedings of the 5th augmented human international conference*, page 15. ACM, 2014.

[64] Julien Pansiot, Benny Lo, and Guang-Zhong Yang. Swimming stroke kinematic analysis with bsn. In *Body Sensor Networks (BSN), 2010 International Conference on*, pages 153–158. IEEE, 2010.

[65] Marc Bächlin and Gerhard Tröster. Swimming performance and technique evaluation with wearable acceleration sensors. *Pervasive and Mobile Computing*, 8(1):68–81, 2012.

[66] GM Lyons, KM Culhane, D Hilton, PA Grace, and D Lyons. A description of an accelerometer-based mobility monitoring technique. *Medical engineering and physics*, 27(6):497–504, 2005.

[67] Wee-Soon Yeoh, Isaac Pek, Yi-Han Yong, Xiang Chen, and Agustinus Borgy Waluyo. Ambulatory monitoring of human posture and walking speed using wearable accelerometer sensors. In *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pages 5184–

5187. IEEE, 2008.

[68] Hristijan Gjoreski, Simon Kozina, Mitja Luštrek, and Matjaž Gams. Using multiple contexts to distinguish standing from sitting with a single accelerometer. In *Proceedings of the Twenty-first European Conference on Artificial Intelligence*, pages 1015–1016. IOS Press, 2014.

[69] Ian Cleland, Basel Kikhia, Chris Nugent, Andrey Boytsov, Josef Hallberg, Kåre Synnes, Sally McClean, and Dewar Finlay. Optimal placement of accelerometers for the detection of everyday activities. *Sensors*, 13(7):9183–9200, 2013.

[70] Jeroen HM Bergmann, Selina Graham, Newton Howard, and Alison McGregor. Comparison of median frequency between traditional and functional sensor placements during activity monitoring. *Measurement*, 46(7):2193–2200, 2013.

[71] Sian E Slawson, Laura M Justham, AA West, PP Conway, MP Caine, and R Harrison. Accelerometer profile recognition of swimming strokes (p17). In *The engineering of sport 7*, pages 81–87. Springer, 2009.

[72] R Delgado-Gonzalo, A Lemkaddem, Ph Renevey, E Muntané Calvo, M Lemay, K Cox, D Ashby, J Willardson, and M Bertschi. Real-time monitoring of swimming performance. In *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the*, pages 4743–4746. IEEE, 2016.

[73] Robert Mooney, Gavin Corley, Alan Godfrey, Leo R Quinlan, and Gearóid ÓLaighin. Inertial sensor technology for elite swimming performance analysis: a systematic review. *Sensors*, 16(1):18, 2015.

[74] Pekka Siirtola, Perttu Laurinen, Juha Röning, and Hannu Kinnunen. Efficient accelerometer-based swimming exercise tracking. In *Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium on*, pages 156–161. IEEE, 2011.

[75] Joe Marshall. Smartphone sensing for distributed swim stroke coaching and research. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous*

computing adjunct publication, pages 1413–1416. ACM, 2013.

[76] Marc Bächlin and Gerhard Tröster. Swimming performance and technique evaluation with wearable acceleration sensors. *Pervasive and Mobile Computing*, 8(1):68–81, 2012.

[77] Julien Pansiot, Benny Lo, and Guang-Zhong Yang. Swimming stroke kinematic analysis with bsn. In *Body Sensor Networks (BSN), 2010 International Conference on*, pages 153–158. IEEE, 2010.

[78] Marc Bächlin, Kilian Förster, and Gerhard Tröster. Swimmaster: a wearable assistant for swimmer. In *Proceedings of the 11th international conference on Ubiquitous computing*, pages 215–224. ACM, 2009.

[79] Yuto Omae, Yoshihisa Kon, Masahiro Kobayashi, Kazuki Sakai, Akira Shionoya, Hirotaka Takahashi, Takuma Akiduki, Kazufumi Nakai, Nobuo Ezaki, Yoshihisa Sakurai, et al. Swimming style classification based on ensemble learning and adaptive feature value by using inertial measurement unit. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 21(4):616–631, 2017.

[80] Woohyeok Choi, Jeungmin Oh, Taiwoo Park, Seongjun Kang, Miri Moon, Uichin Lee, Inseok Hwang, and Junehwa Song. Mobydick: an interactive multi-swimmer exergame. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, pages 76–90. ACM, 2014.

[81] Jeroen Lecoutere and Robert Puers. Wireless communication with miniaturized sensor devices in swimming. *Procedia Engineering*, 72:398–403, 2014.

[82] S Slawson, P Conway, L Justham, T Le Sage, and A West. Dynamic signature for tumble turn performance in swimming. *Procedia Engineering*, 2(2):3391–3396, 2010.

[83] Ilkka Keskinen. Listen to your eyes, learning swimming technique, 2000.

[84] Neil P Davey, Megan E Anderson, and Daniel A James. An accelerometer-based

system for elite athlete swimming performance analysis. In *Smart Structures, Devices, and Systems II*, volume 5649, pages 409–416. International Society for Optics and Photonics, 2005.

[85] How-Lung Eng, Kar-Ann Toh, Wei-Yun Yau, and Junxian Wang. Dews: A live visual surveillance system for early drowning detection at pool. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(2):196–210, 2008.

[86] Meng-Shiuan Pan, Ku-Chen Huang, Tzu-Hsiu Lu, and Ze-Yan Lin. Using accelerometer for counting and identifying swimming strokes. *Pervasive and Mobile Computing*, 31:37–49, 2016.

[87] Marc Bächlin and Gerhard Tröster. Pervasive computing in swimming: A model describing acceleration data of body worn sensors in crawl swimming. In *Pervasive Computing (JCPC), 2009 Joint Conferences on*, pages 293–298. IEEE, 2009.

[88] Marko Topalovic, Simon Eyers, Vasileios Exadaktylos, Jan Olbrecht, Daniel Berckmans, and Jean-Marie Aerts. Online monitoring of swimmer training using a 3d accelerometer-identifying swimming and swimming style. In *Proceedings of the 2nd International Congress on Sports Sciences Research and Technology Support*, pages 111–115. SCITEPRESS–Science and Technology Publications, 2014.

[89] Yoshihisa Kon, Yuto Omae, Kazuki Sakai, Hirotaka Takahashi, Takuma Akiduki, Chikara Miyaji, Yoshihisa Sakurai, Nobuo Ezaki, and Kazufumi Nakai. Toward classification of swimming style by using underwater wireless accelerometer data. In *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*, pages 85–88. ACM, 2015.

[90] Josh Cherian, Vijay Rajanna, Daniel Goldberg, and Tracy Hammond. Did you remember to brush?: a noninvasive wearable approach to recognizing brushing teeth for elderly care. In *Proceedings of the 11th EAI International Conference*

on Pervasive Computing Technologies for Healthcare, pages 48–57. ACM, 2017.

[91] Dong Xiao, Zhiwen Yu, Fei Yi, Liang Wang, Chiu C Tan, and Bin Guo. Smartswim: An infrastructure-free swimmer localization system based on smartphone sensors. In *International Conference on Smart Homes and Health Telematics*, pages 222–234. Springer, 2016.

[92] Ferhat Attal, Samer Mohammed, Mariam Dedabrishvili, Faicel Chamroukhi, Latifa Oukhellou, and Yacine Amirat. Physical human activity recognition using wearable sensors. *Sensors*, 15(12):31314–31338, 2015.

[93] Danielle Cummings, George Lucchese, Manoj Prasad, Chris Aikens, Jimmy Ho, and Tracy Hammond. Haptic and ar interface for paratrooper coordination. In *Proc. of 13th Int. Conf. of NZ Chapter of the ACM's Special Interest Group on HCI*, pages 52–55. ACM, 2012.

[94] Danielle Cummings, George Lucchese, Manoj Prasad, Christopher Aikens, Jimmy Ho, and Tracy Hammond. GeoTrooper: a mobile location-aware system for team coordination. In *Proceedings of the 13th International Conference of the NZ Chapter of the ACM's Special Interest Group on Human-Computer Interaction*, page 102. ACM, 2012.

[95] Danielle Cummings, Manoj Prasad, George Lucchese, Christopher Aikens, and Tracy Hammond. Multi-modal location-aware system for paratrooper team coordination. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*, pages 2385–2388. ACM, 2013.

[96] George Lucchese, Manoj Prasad, Danielle Cummings, Chris Aikens, Jimmy Ho, and Tracy Hammond. Geotrooper a location-aware system for rapid team assembly. In *Texas A&M University Industrial Affiliates Program (IAP)*, College Station, TX, USA, March 10, 2011.

[97] Manoj Prasad, Paul Taele, Daniel Goldberg, and Tracy A Hammond. Haptimoto:

Turn-by-turn haptic route guidance interface for motorcyclists. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 3597–3606. ACM, 2014.

[98] Emily Berk, Paul Taele, Stephanie Valentine, and Tracy Hammond. Haptic feedback from wearable device for indoor navigation in disaster situations. In *Proceedings of Texas A&M University Undergraduate Research Summer Poster Session*, College Station, TX, USA, August 8, 2014. Winner, Best Poster.

[99] NeerajYadav. Warning system for outdoor construction workers using haptic communication, proposal. Master's thesis, Texas A&M University, December 2016.

[100] J. Byun, D.W. Goldberg, T.A. Hammond, and J. Horney. Investigating navigational wearables for people living with dementia. In *Proceedings of Texas A&M University Undergraduate Research Summer Poster Session*, College Station, TX, USA, August 3, 2016.

[101] Manoj Prasad, Paul Taele, Ayobami Olubeko, and Tracy Hammond. Haptigo: A navigational tap on the shoulder. In *Haptics Symposium (HAPTICS), 2014 IEEE*, pages 339–345, Feb 2014.

[102] Tracy Hammond, Manoj Prasad, Ayobami Olubeko, and Paul Taele. Haptigo: an intelligent and lightweight tactile vest for improving active pedestrian navigation experiences. In *ACM Richard Tapia Celebration of Diversity in Computing (TAPIA)*, Washington, D.C., February 07, 2013.

[103] David Brhlik, Temiloluwa Otuyelu, and Chad Young. Enhancing blind navigation with the use of wearable sensor technology. Undergraduate honors thesis, Texas A&M University, May 2016.

[104] Temiloluwa Otuyelu. Enhancing blind navigation with the use of wearable sensor technology. Undergraduate honors thesis, Texas A&M University, May 2016.

[105] Chad Young. Enhancing blind navigation with the use of wearable sensor

technology. Undergraduate honors thesis, Texas A&M University, May 2016.

[106] Chelsea Hall, Stefan Levy, Keleigh Maddox, Brett Walker, Christine Woods, and Tracy Hammond. Vibro-tactile pattern perception. In *Texas A&M University Student Research Week*, College Station, TX, USA, March 24, 2014.

[107] Nahum Villanueva Luna. ARCaching: Using augmented reality on mobile devices to improve geocachers' experiences. Master's thesis, Texas A&M University, May 2017.

[108] Manoj Prasad. *Designing tactile interfaces for abstract interpersonal communication, pedestrian navigation and motorcyclists navigation*. PhD thesis, Texas A&M University, College Station, Texas, 5 2014. 183 pages.

[109] Manoj Prasad, Murat Russell, and Tracy Hammond. Vibro-tactile pattern perception. In *Texas A&M University Industrial Affiliates Program (IAP)*, College Station, TX, USA, March 6, 2012.

[110] Murat Russell, Manoj Prasad, and Tracy Hammond. The haptic pattern transducer. In *Texas A&M University Industrial Affiliates Program (IAP)*, College Station, TX, USA, March 6, 2012.

APPENDIX

Table A.1: Flip turns cross validation on traditional features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	75.0	79.310	83.222	85.731	81.843	82.288	81.856	80.705
Naive Bayes	79.688	77.586	78.667	79.599	75.821	76.102	76.564	76.685
Decision Stump	81.25	81.034	76.0	76.769	74.544	73.644	71.134	71.614
J48 Decision Tree	79.688	79.310	89.889	89.033	87.591	88.22	84.948	91.218
Random Forest Tree	79.688	82.758	92.111	93.632	92.062	92.627	91.34	94.805
Random Decision Tree	71.875	72.413	88.111	90.566	89.507	88.644	88.729	91.28
Repetitive Decision Tree	79.685	84.482	85.444	87.854	86.223	85.085	85.842	89.054
Nearest Neighbor	79.687	84.482	85.444	87.854	86.223	85.085	85.842	89.054

Table A.2: Flip turns independent validation on traditional features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	85.934	85.662	88.485	81.286	78.214	75.854	73.763	73.995
Naive Bayes	85.687	83.611	88.159	84.433	81.206	81.384	75.054	75.484
Decision Stump	88.661	93.472	82.556	77.822	78.489	76.392	58.755	68.127
J48 Decision Tree	86.126	84.274	87.203	84.891	76.566	83.657	73.221	76.879
Random Forest Tree	85.357	86.325	83.784	83.667	72.643	81.399	75.945	78.689
Random Decision Tree	82.857	84.274	79.254	86.689	68.245	75.902	72.894	75.148
Repetitive Decision Tree	88.434	87.094	91.577	79.68	78.489	80.63	73.446	79.222
Nearest Neighbor	88.434	87.094	91.577	79.68	78.489	80.63	73.446	79.222

Table A.3: Flip turns cross validation on novel features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	63.793	65.625	82.23	77.829	79.166	75.989	74.145	74.038
Naive Bayes	65.517	68.75	45.588	43.533	42.234	37.994	34.33	39.807
Decision Stump	72.414	78.125	50.98	50.808	50.189	50.043	48.148	50.192
J48 Decision Tree	74.138	76.562	87.132	88.222	86.268	84.696	83.831	81.538
Random Forest Tree	74.138	54.688	96.078	97.46	96.022	95.162	93.874	94.23
Random Decision Tree	72.414	54.688	82.23	85.334	83.049	83.553	81.766	82.371
Repetitive Decision Tree	65.517	73.438	78.921	81.409	76.231	78.54	75.569	75.512
Nearest Neighbor	65.517	73.438	78.921	81.409	76.231	78.54	75.569	75.512

Table A.4: Flip turns independent validation on novel features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	67.793	74.744	77.117	77.949	67.781	71.213	66.702	66.673
Naive Bayes	61.062	67.222	74.04	73.203	67.012	57.648	64.613	65.747
Decision Stump	83.187	83.803	76.861	65.196	67.381	67.636	71.591	66.537
J48 Decision Tree	76.071	82.521	78.399	71.279	77.921	64.617	72.446	66.537
Random Forest Tree	80.11	62.906	64.211	79.744	69.655	73.328	60.542	58.51
Random Decision Tree	69.148	62.906	64.211	72.39	63.443	68.186	60.542	58.51
Repetitive Decision Tree	69.588	73.547	69.425	73.311	70.888	65.695	71.084	71.53
Nearest Neighbor	69.588	73.547	69.425	73.311	70.888	65.695	71.084	71.53

Table A.5: Flip turns cross validation on traditional and novel features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	94.444	95.0	86.397	92.148	84.659	85.664	88.604	88.462
Naive Bayes	75.926	78.333	72.304	74.365	72.348	73.351	66.311	70.0
Decision Stump	77.778	85.0	74.142	74.365	72.443	72.032	69.801	70.705
J48 Decision Tree	88.889	91.667	91.176	91.686	88.92	91.557	91.667	93.718
Random Forest Tree	96.296	100.0	96.201	95.612	94.508	96.482	95.94	97.051
Random Decision Tree	96.296	98.333	91.422	91.57	89.678	93.14	90.527	92.372
Repetitive Decision Tree	88.889	86.667	90.319	88.568	85.985	88.918	89.103	91.538
Nearest Neighbor	88.889	86.667	90.319	88.568	85.985	88.918	89.103	91.538

Table A.6: Flip turns independent validation on traditional and novel features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	89.494	75.208	91.818	81.267	85.04	84.559	83.877	86.411
Naive Bayes	83.72	85.694	86.077	89.6	76.617	82.337	65.593	75.789
Decision Stump	88.661	93.472	88.325	90.283	77.153	78.268	68.127	58.755
J48 Decision Tree	85.923	78.542	77.214	82.709	84.544	82.598	80.158	76.162
Random Forest Tree	92.827	93.472	87.542	89.898	88.71	87.042	88.502	82.028
Random Decision Tree	83.105	84.583	79.714	83.666	80.109	76.258	77.567	78.412
Repetitive Decision Tree	81.161	82.917	91.355	87.793	80.377	73.284	75.859	75.917
Nearest Neighbor	81.161	82.917	91.355	87.793	80.377	73.284	75.859	75.917

Table A.7: Combined flip turns and swim strokes cross validation using traditional features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	93.977	92.503	98.635	98.345	97.745	97.866	96.566	96.26
Naive Bayes	90.59	90.773	94.353	94.369	93.738	93.658	92.091	92.089
Decision Stump	35.508	39.446	35.592	35.061	34.93	34.906	33.614	33.994
J48 Decision Tree	90.59	91.119	98.449	98.675	98.102	98.139	96.987	97.137
Random Forest Tree	94.103	94.464	99.503	99.474	99.108	99.182	98.622	98.687
Random Decision Tree	88.457	91.003	98.234	98.443	97.847	97.837	96.813	96.81
Repetitive Decision Tree	89.21	89.619	97.915	98.114	97.203	97.238	96.487	96.27
Nearest Neighbor	89.21	89.619	97.915	98.114	97.203	97.238	96.487	96.27

Table A.8: Combined flip turns and swim strokes independent validation using traditional features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	83.989	82.554	82.892	84.429	81.611	83.322	80.877	82.991
Naive Bayes	84.135	82.658	82.904	82.59	82.386	82.904	82.057	81.431
Decision Stump	23.472	27.247	25.711	25.378	25.25	26.675	24.962	22.04
J48 Decision Tree	80.812	79.706	78.958	79.087	80.127	74.879	81.917	77.891
Random Forest Tree	86.188	87.102	85.653	87.035	84.569	85.625	83.946	84.447
Random Decision Tree	80.2	76.044	78.067	77.792	77.447	75.005	73.341	67.46
Repetitive Decision Tree	84.009	80.655	80.201	83.309	77.438	73.386	80.904	78.046
Nearest Neighbor	84.009	80.655	80.201	83.309	77.438	73.386	80.904	78.046

Table A.9: Combined flip turns and swim strokes cross validation on novel features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	80.979	89.273	95.629	96.571	95.751	95.703	94.094	94.355
Naive Bayes	74.998	84.198	88.898	88.584	88.329	88.163	86.648	86.676
Decision Stump	25.576	39.331	40.928	40.867	40.243	40.258	39.567	39.511
J48 Decision Tree	76.903	88.351	97.358	97.567	96.859	96.859	95.809	96.142
Random Forest Tree	83.131	91.811	98.939	99.032	98.573	98.672	98.054	98.17
Random Decision Tree	72.298	85.121	96.764	96.746	96.592	96.444	94.789	95.237
Repetitive Decision Tree	76.231	87.889	96.445	96.403	95.713	95.886	94.904	95.184
Nearest Neighbor	76.231	87.889	96.445	96.403	95.713	95.886	94.904	95.184

Table A.10: Combined flip turns and swim strokes independent validation on novel features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	78.081	80.979	80.069	78.841	79.707	77.486	75.858	80.134
Naive Bayes	75.324	74.998	74.446	73.512	73.926	71.904	71.684	74.325
Decision Stump	24.295	25.576	26.105	24.8	26.496	26.43	24.894	21.386
J48 Decision Tree	74.522	76.903	73.273	74.605	72.22	74.843	76.331	73.855
Random Forest Tree	82.943	83.131	81.903	84.163	82.27	83.258	81.215	81.154
Random Decision Tree	74.305	72.298	69.756	70.53	73.322	71.144	67.766	71.724
Repetitive Decision Tree	80.5	76.231	76.249	74.581	73.832	75.168	73.215	76.403
Nearest Neighbor	80.5	76.231	76.249	74.581	73.832	75.168	73.215	76.403

Table A.11: Combined flip turns and swim strokes cross validation on traditional and novel features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	96.487	96.77	98.575	98.647	97.541	97.955	96.247	96.461
Naive Bayes	89.711	89.504	94.13	93.935	93.273	93.041	90.755	90.59
Decision Stump	33.877	34.025	35.421	34.991	34.975	34.9	33.795	34.307
J48 Decision Tree	97.616	96.655	98.494	98.591	98.095	97.931	96.748	97.012
Random Forest Tree	98.62	98.731	99.391	99.439	99.044	99.146	98.424	98.432
Random Decision Tree	94.228	94.233	97.929	98.065	97.445	97.297	96.216	96.675
Repetitive Decision Tree	94.228	94.348	97.989	97.981	97.324	97.285	96.01	96.428
Nearest Neighbor	94.228	94.348	97.989	97.981	97.324	97.285	96.01	96.428

Table A.12: Combined flip turns and swim strokes independent validation on traditional and novel features

Window Size (ms)	3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250
Multilayer Perceptron	86.209	88.032	87.714	86.158	84.997	84.92	85.778	83.767
Naive Bayes	81.013	80.783	78.85	78.035	78.223	78.463	79.002	79.258
Decision Stump	21.855	22.168	21.966	21.984	21.735	21.705	24.782	21.935
J48 Decision Tree	81.349	76.292	78.828	75.912	78.361	78.729	83.252	78.389
Random Forest Tree	86.261	86.628	86.231	85.845	86.214	85.912	86.567	87.004
Random Decision Tree	76.686	78.841	80.12	74.894	76.825	77.253	77.982	77.821
Repetitive Decision Tree	79.006	80.137	81.612	80.756	78.523	77.999	76.834	83.107
Nearest Neighbor	79.006	80.137	81.612	80.756	78.523	77.999	76.834	83.107

Table A.13: Major swim styles cross validation on traditional features

Window Size (ms)	4000		3500		3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250	500	250	500	250
Multilayer Perceptron	96.03	96.964	96.26	96.842	95.264	95.392	96.685	97.448	95.858	96.064	95.65	96.015
Naive Bayes	93.573	93.036	93.171	93.835	93.911	93.151	92.707	92.836	93.491	93.222	93.564	91.023
Decision Stump	43.1	43.571	40.325	42.556	42.219	40.847	42.099	42.198	41.927	41.983	41.597	41.575
J48 Decision Tree	93.195	94.107	96.26	95.789	93.369	94.396	94.033	95.976	93.66	94.606	93.981	95.199
Random Forest Tree	97.164	97.857	97.886	97.594	96.752	96.762	97.127	97.645	96.365	97.303	96.484	96.88
Random Decision Tree	94.896	94.107	94.959	94.586	94.452	93.275	92.818	94.701	93.322	94.534	93.981	94.095
Repetitive Decision Tree	94.14	94.286	95.447	94.586	94.993	94.396	94.696	94.603	95.097	95.044	94.994	95.247
Nearest Neighbor	94.14	94.286	95.447	94.586	94.993	94.396	94.696	94.603	95.097	95.044	94.994	95.247

Table A.14: Major swim styles independent validation on traditional features

Window Size (ms)	4000		3500		3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250	500	250	500	250
Multilayer Perceptron	89.959	88.703	87.674	88.095	89.427	87.695	89.031	89.881	87.009	89.16	87.692	89.468
Naive Bayes	88.939	88.857	87.79	88.987	87.396	85.361	87.737	88.11	88.408	88.998	88.14	85.144
Decision Stump	30.92	31.134	25.921	30.269	29.85	30.216	29.99	29.494	28.312	27.166	27.054	28.177
J48 Decision Tree	83.463	80.803	80.663	81.9	75.754	80.358	80.755	83.429	79.907	79.872	86.193	82.231
Random Forest Tree	89.76	90.137	88.33	88.349	87.456	88.456	87.824	89.51	88.724	87.885	88.615	87.49
Random Decision Tree	83.209	79.401	81.056	81.549	83.027	82.753	80.755	83.175	83.546	77.411	85.344	84.942
Repetitive Decision Tree	86.551	81.618	87.63	88.919	85.368	88.519	84.896	81.917	86.257	83.932	86.563	84.321
Nearest Neighbor	86.551	81.618	87.63	88.919	85.368	88.519	84.896	81.917	86.257	83.932	86.563	84.321

Table A.15: Major swim styles cross validation on novel features

Window Size (ms)	4000		3500		3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250	500	250	500	250
Multilayer Perceptron	99.023	98.809	98.754	99.156	98.513	98.649	98.564	98.418	98.016	98.64	97.848	98.003
Naive Bayes	93.422	93.556	93.749	93.746	93.785	93.645	93.628	93.686	93.17	93.121	91.666	91.649
Decision Stump	44.784	44.736	44.407	44.379	44.032	44.034	43.763	43.712	43.359	43.374	38.583	41.258
J48 Decision Tree	99.539	99.319	99.272	99.434	98.818	99.076	98.889	98.958	98.696	98.87	98.373	98.497
Random Forest Tree	99.78	99.777	99.721	99.655	99.66	99.709	99.611	99.573	99.506	99.54	99.246	99.331
Random Decision Tree	99.121	98.926	98.833	99.031	98.988	98.734	98.334	98.59	98.442	98.697	97.928	97.957
Repetitive Decision Tree	99.22	99.075	98.853	99.079	98.621	98.7	98.619	98.358	98.696	98.435	98.048	98.014
Nearest Neighbor	99.22	99.075	98.853	99.079	98.621	98.7	98.619	98.358	98.696	98.435	98.048	98.014

Table A.16: Major swim styles independent validation on novel features

Window Size (ms)	4000		3500		3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250	500	250	500	250
Multilayer Perceptron	90.107	89.024	86.483	86.857	89.146	89.868	88.635	87.735	88.459	86.369	87.974	88.126
Naive Bayes	81.291	80.144	81.551	80.097	79.062	81.597	82.041	80.999	81.11	79.676	80.988	80.793
Decision Stump	28.357	31.134	27.32	28.459	26.565	28.881	28.18	28.213	22.104	28.645	22.866	19.469
J48 Decision Tree	82.726	83.021	77.471	83.003	80.812	82.967	82.123	84.057	83.917	81.549	82.188	84.522
Random Forest Tree	86.325	85.49	87.883	88.433	87.509	88.469	86.971	89.391	86.857	87.772	88.375	88.074
Random Decision Tree	82.205	81.56	78.063	79.302	83.174	83.351	80.06	81.696	83.162	81.051	82.465	81.796
Repetitive Decision Tree	84.641	80.086	80.239	82.356	80.262	83.186	84.976	84.24	82.203	83.173	86.217	83.477
Nearest Neighbor	84.641	80.086	80.239	82.356	80.262	83.186	84.976	84.24	82.203	83.173	86.217	83.477

Table A.17: Major swim styles cross validation on traditional and novel features

Window Size (ms)	4000		3500		3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250	500	250	500	250
Multilayer Perceptron	99.703	99.724	99.541	99.568	99.481	99.444	99.279	99.425	99.492	99.565	99.184	99.228
Naive Bayes	97.046	96.959	96.68	97.065	96.857	96.741	96.142	96.27	97.032	96.678	95.507	95.621
Decision Stump	44.784	44.736	44.407	39.363	44.032	44.034	42.536	42.888	37.669	43.652	41.472	35.711
J48 Decision Tree	99.649	99.713	99.601	99.53	99.463	99.521	99.396	99.419	99.445	99.468	99.144	99.069
Random Forest Tree	99.912	99.904	99.741	99.789	99.803	99.743	99.725	99.649	99.77	99.79	99.435	99.552
Random Decision Tree	99.583	99.479	99.043	99.434	99.051	99.128	98.799	98.927	98.992	99.07	98.476	98.755
Repetitive Decision Tree	99.605	99.49	99.302	99.357	99.31	99.299	99.08	99.189	99.191	99.243	98.898	98.914
Nearest Neighbor	99.605	99.49	99.302	99.357	99.31	99.299	99.08	99.189	99.191	99.243	98.898	98.914

Table A.18: Major swim styles independent validation on novel features

Window Size (ms)	4000		3500		3000		2500		2000		1500	
Window Overlap (ms)	500	250	500	250	500	250	500	250	500	250	500	250
Multilayer Perceptron	90.566	88.435	91.707	90.932	89.343	92.263	90.249	89.365	90.444	89.791	90.362	87.781
Naive Bayes	89.513	88.697	86.781	88.419	85.565	86.394	88.39	85.887	90.061	83.609	85.338	85.006
Decision Stump	26.976	29.749	23.701	24.19	22.75	23.676	27.212	24.35	42.43	30.114	28.372	24.937
J48 Decision Tree	81.119	86.477	81.848	75.757	85.923	82.094	78.253	83.143	84.24	86.502	83.972	85.156
Random Forest Tree	91.381	89.59	90.447	90.821	90.048	89.175	88.021	89.887	87.641	88.267	90.442	90.38
Random Decision Tree	79.289	80.197	81.799	83.537	85.978	81.126	80.889	82.424	83.141	79.971	85.089	87.258
Repetitive Decision Tree	81.826	84.498	84.353	77.321	85.79	83.324	81.028	82.759	86.998	84.338	83.602	89.967
Nearest Neighbor	81.826	84.498	84.353	77.321	85.79	83.324	81.028	82.759	86.998	84.338	83.602	89.967