Wearable Sensors in Extreme Environments

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Thomas Beckingham

June 2023

Acknowledgements

I would like to start by thanking the people who have supported me during this research.

Dr Kirsty McKay, my supervisor not only during this PhD, but also my undergraduate degree. I am extremely appreciative of the time you have invested in me and for convincing me to do a PhD in the first place!

Professor Joe Spencer, my secondary supervisor during this research, thank you very much for your support and introduction into the world of sensors and the analysis of sensor data.

My family, friends and girlfriend for all of your support during this work without which these four years would've been impossible.

The University of Liverpool EEE workshop staff who have helped/saved me more times than I care to admit!

And to the countless others who have assisted me during the course of this research. Thank you.

Abstract

Demonstrated within this thesis is the use of sensors in extreme environments which is an important field of study considering the special nature associated with such deployments. The issue being that often extreme environments present conditions that either inhibit sensor performance or place humans under increased strain/risk. Hence, the main aim of this thesis is to investigate sensing applications in scenarios that demonstrate the aforementioned difficulties. Specifically, this work focuses on the construction of a wearable sensor platform that is capable of deployment into multiple extreme environments without the need for significant electronic modification. To minimise cost, the system has been constructed using off the shelf components instead of specialised devices with focus directed towards signal/data processing instead of sensor fabrication. The developed algorithms are used to both interpret the raw signal data and then process this information into metrics relative to the deployment environment(s), which for this thesis are microgravity and underwater. Key algorithms/methods included within this work include: neural networks; machine learning; signal processing techniques; artefact replication and use of feature generation techniques.

Key findings included in this work include novel methods for sleep stage prediction via both logical and clustering approaches, and the results achieved when applied to multiple participants. In addition, a new method for the analysis of underwater ECG signals via use of a neural network is developed. Crucial to this approach is the concept of artefact replication, mentioned above, which potentially has multiple additional applications not limited to swim analysis. Also, new swim stroke classification methods are detailed using accelerometer data sampled from the chest region, and subsequent application of feature generation techniques to identify stroke. The identified swim stroke can then be used to estimate breathing rate which is another important metric relative to swimming. Interestingly, the feature generation methods used here can also improve the performance of the sleep stage analysis methods. Also, the use of WiFi for underwater data transmission is investigated and the limitations of the specific use case in this thesis have been explored. Generally, good performance is achieved by all algorithms developed as part of this thesis. In particular, the BPM estimation method designed for application to underwater ECG samples, which achieves RMSE: 0.58 for counted underwater heartbeats (published in journal: Applied Intelligence). Key conclusions drawn from this work include the feasibility

of developing sensor platforms intended for deployment into extreme environments, with emphasis placed on signal/data processing instead of the use/fabrication of specialised sensors. Also highlighted, is the versatility of using a common set of sensors, to generate a range of physiological metrics relative to a given deployment environment.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis explores the use of wearable sensors deployed in extreme environments and how applying signal/data processing techniques and algorithms can allow useful information to be generated from raw data sampled in such circumstances. Specifically, the sensors record various physiological metrics which are highlighted in subsequent sections. Prior to investigating the use of sensors in extreme environments, it is first important to define what is considered an extreme environment relative to this thesis. For such applications, an environment is considered extreme if it presents conditions that either inhibit sensor performance or place humans operating in said environment under increased strain/risk. The two environments investigated in this thesis are microgravity and underwater which provide assessment of the two distinctions highlighted above. The microgravity environment is considered extreme specific to the human perspective whereas the underwater environment is extreme for both the sensors and humans. Hence, investigating these two scenarios allows understanding of both aspects. Also, the two environments/applications provide an understanding of the sensitivity of the sensor platform considering the minor physiological changes associated with different sleep stages compared to the larger physiological

changes associated with underwater activity (swimming for example). For the microgravity investigation, novel sleep stage analysis algorithms are developed that interpret the recorded physiological data and attempt to predict the appropriate sleep stages. This is important considering that astronauts often report poor sleep quality in space hence the techniques developed are intended to provide a lightweight diagnostic tool that will help better understand sleep in microgravity. Considering the difficulties associated with performing experimentation in space, testing thus far has been conducted on Earth. However, the design of the sensor system is such that it attempts to realise the constraints associated with deployment into space. Also, this work has been presented at the International Aeronautical Conference 2019 [1]. For the underwater application, various swim analysis algorithms have been developed that are intended for use by recreational/professional swimmers to understand swim performance. Thus far, all underwater experimentation has been performed in a bath or swimming pool however extension to open water swimming and search and rescue applications is discussed. Notably, a method for predicting heart rate from noisy underwater ECG samples has been devised and this work has been published in the journal: Applied Intelligence [2]. The remainder of this chapter serves to introduce this thesis by first specifying the research aims and objectives, explaining the principles of key sensor technology used in this work and finally discussing the literature on sensors in extreme environments.

1.1 Aims & Objectives

1.1.1 Aims

The main aim of this thesis is the development of a lightweight, wearable sensor platform capable of deployment into multiple different extreme environments without needing major

electronic modification. Such a system was required to have the capability for wireless transmission and analysis hence enabling real time monitoring/telemedicine capabilities. In addition, the sensor system should be inexpensive and easy to distribute.

1.1.2 Objectives

To achieve the main aim of this thesis, a robust wearable sensor platform was designed and constructed that is capable of functioning in multiple extreme environments. This device will measure physiological quantities including: heart rate; breathing rate; skin temperature; body acceleration and process this information into quantities relative to the deployment environment. Namely, sleep stage prediction and swim analysis metrics (swim stroke classification/underwater heart rate estimation). For classification tasks, at least 90% accuracy was the target value compared to regression tasks where less than 0.5 RMSE was the desired score. The cost target for the sensor platform was less than £100. Considering the real time focus of this work, the system/algorithms were designed to provide updates at least every 10 seconds. The wireless transmission capability required for this work was reliable signal extraction when using the device particularly whilst underwater.

1.2 Wearable sensor technology

Wearable sensors are devices designed to be worn on/near an individual monitoring various physical and physiological parameters. Such sensors have gained significant attention in recent years due to the rapid advancements in wearable technology and the growing demand for non-invasive methods of monitoring health and wellness. Wearable sensors can measure a wide range of parameters including heart rate, temperature, body movement etc. One of the earliest forms of wearable sensors was the heart rate monitor, which was first introduced in the 1957 [3]. This was the first portable ECG monitoring device, how-

ever ECGs had been recorded since 1902 [4]. Since then, wearable sensor technology has advanced significantly and now includes devices such as smartwatches, activity trackers and sleep monitors. Overall, wearable sensors are becoming increasingly popular due to their non-invasive nature, ease of use, and ability to provide real-time data about a variety of physical and physiological parameters. As technology continues to advance, it is likely that wearable sensors will become even more prevalent and integrated into society.

Wearable sensor technology have many applications, two of particular interest to this thesis being: health care and sport monitoring. Advances in wearable technology can benefit the health care system in multiple ways such as: diagnostic; rehabilitation and remote monitoring capabilities [5]. This progress is not solely due to development of sensor technology but is rather is the result of the advances in information technology and related fields as a whole [5]. As stated, the advances to wearable health care technology often benefit sporting applications. For example, accurate monitoring of heart rate is an important health diagnostic in medical settings, it is also useful in sporting applications and can be used to determine how many calories a subject has burned [6, 7].

Wearable sensors commonly used to understand the human body include:

- Electrocardiogram (ECG) sensors: purpose of which is to monitor the biopotential of the heart which in turn allows the generation of vital metrics such a heart Beats Per Minute (BPM). Example output from an ECG sensor is demonstrated by figure 1.1. This demonstrates the electrical trace of the heart for one heart beat, the consistent components of the ECG waveform (P, Q, R, S and T waves) are labelled. To generate such a sample, an electrical amplification configuration must be used to amplify the small electrical signals measured via chest electrodes. Use of a differential amplifier, operational amplifier and additional signal filtering to the chest electrode signal generates an ECG trace [8]. The ECG trace can be used to calculate BPM or

diagnose cardiac disease [9].

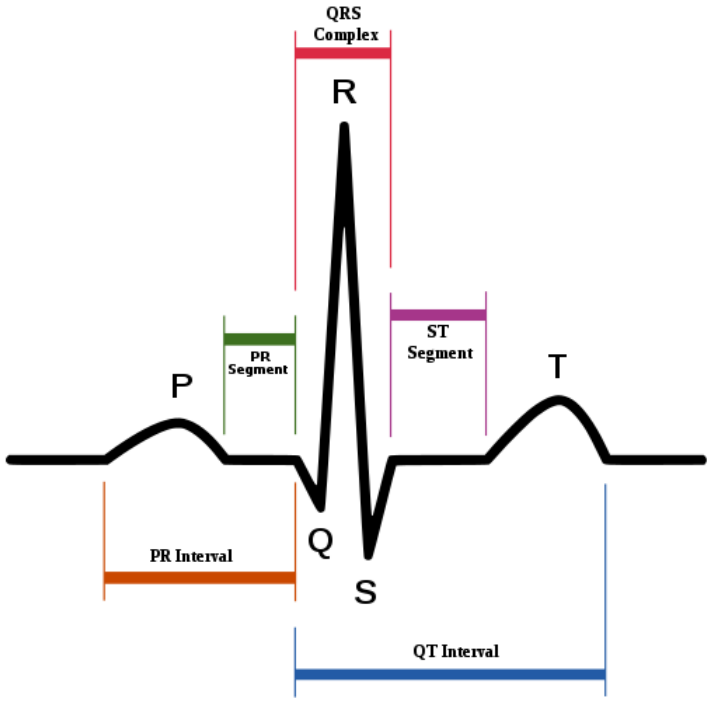


Figure 1.1: ECG sample labelled with P, Q, R, S & T waveforms. Taken from [10].

- Electromyography (EMG) sensors: similar to ECG sensors, EMG devices also amplify biopotentials. However, instead of amplifying the electrical signals of the heart, EMG sensors amplify the electrical signals of the muscles; bicep contractions for example. An EMG sensor also uses an amplifier configuration applied to electrode signals in order to generate usable data. Use of an ECG sensor for the purpose of EMG measurement is not possible considering the voltage output range and signal bandwidth of larger muscle contractions, is different compared to the heart. Hence, a separate electronic configuration is necessary [11]. Applications of EMG include assessing muscle contraction strength post spinal injury [12].

- Electroencelaphogram (EEG) sensors: used to monitor the electrical activity of the

brain. Similar to ECG and EMG this involves amplification of electrical signals measured from electrodes. EEG sensors can be used for sleep analysis purposes to identify what sleep stage a subject is in based upon brain activity [13]. Figure 1.2 demonstrates how different EEG wave patterns can be used to identify the individual sleep stages.

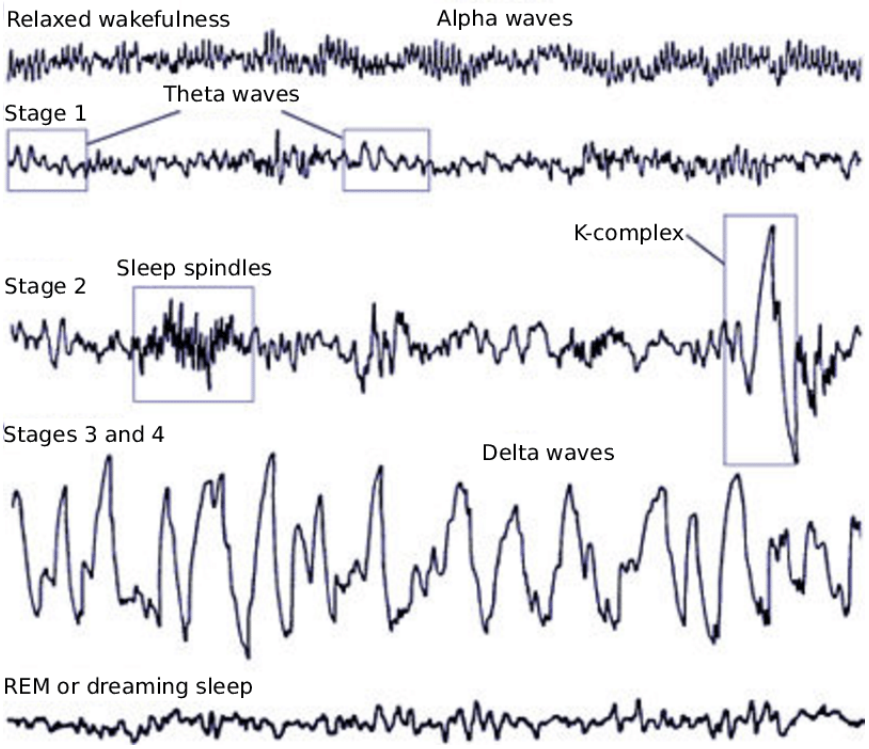


Figure 1.2: Labelled EEG data highlighting the wave components associated with each sleep stage. Taken from [14]

- Electrodermal Activity (EDA) sensors, also known as Galvanic Skin Response (GSR) are devices used to measure the electrical conductance of the skin. These sensors are commonly used in various fields such as psychology, neuroscience, and human-computer interaction to assess the physiological/emotional state of an individual [15]. These sensors work by detecting changes in the electrical properties of the

skin, specifically the conductivity or resistance of the sweat glands. When a person experiences emotional or physiological stimulation, such as excitement or stress the activity of the sweat glands increases, leading to changes in skin conductance [15]. Such devices work via use of electrodes that are placed on the skin, usually on the fingers or palms. One electrode is used to deliver a small electrical current to the skin, while the other electrode measures the resulting electrical conductance [15].

- Photoplethysmography (PPG) sensors: are an optical approach for analysis of the heart. PPG uses a light source and a photodetector at the surface of skin that measures reflected light [16]. This indicates the volumetric variations of blood circulation which in turn indicates the BPM [16]. Similar to ECG, PPG is used for cardiac analysis and is commonly employed by smart watches to calculate BPM during exercise [16]. However, PPG has limitations such as inaccuracies especially at heightened BPM ranges [17].

- Breathing rate sensors: monitoring of respiration rate is commonly performed via use of elasticated chest strap whereby the expansion of the chest alters some electrical property. For example, inductance plethysmography can be used to relate a change in inductance to the expansion and contraction of the chest [18]. A similar process may be performed, however instead of measuring inductance, assessing a change in resistance that relates to breathing rate. Alternatively, respiration rate may be estimated via use of a triple axis accelorometer, situated in the chest region to record breathing related chest movements [19]. Such an approach is obviously highly susceptible to motion artefacts however. A solution to this involves assessment of the ECG trace from which breathing rate can be identified from the R height peak modulation [20]. However, this approach is hindered by any noise present on the ECG waveform as is the case for multiple applications within this thesis.

- Temperature sensors: can be used for analysis of body temperature, in particular core body temperature, which indicates the temperature of internal organs [21]. Thermoregulation is a key process in normal bodily function that maintains stable bodily temperatures regardless of external conditions [21]. Accurate measurement of core body temperature can be invasive (rectal). However investigations into non-invasive measurements via combined use of: skin temperature; heat flux and heart rate, at specific sites around the body (rib cage, pectoralis, inner thigh etc) have demonstrated good core body temperature predictions [22].

- Oxygen saturation (SP02) sensors: are used to measure how much oxygen is present in the blood [23]. In sport settings, SP02 can be employed as a metric used to assess an individuals 'maximum effort', which can help identify an athlete's fitness and health status [24]. To measure SP02, the skin is illuminated, commonly with red and infrared light. The changes in light absorption, caused by the presence oxygenated and de-oxygenated blood, can be measured from which SP02 is deduced [23].

- Movement sensors: assessment of movement is important in dynamic situations particularly in sports applications [25]. This is because accelerometers, for example, provide an understanding of the accelerations an athlete/individual undergoes. The use of a gyroscope may also be beneficial for assessing movement to understand the rotations associated with activity. An Inertial Measurement Unit (IMU) provides both acceleration and rotational information by combining an accelerometer and gyroscope to create a 6 (3 axis accelerometer 3 axis and gyroscope) Degree Of Freedom (DOF) device [26]. Data fusion can be applied to the data streams to provide a better understanding of motion [27]. In addition, a magnometer can be included yielding a 9 DOF IMU so to include magnetic heading information [27].

1.2.1 Sleep section stage analysis

An example relevant to this thesis, of multiple biometric used in unison to generate new physiological metrics, is sleep analysis. Sleep is an issue for many humans who may suffer from disorders such as: insomnia [28], narcolepsy [29]; apnoea [30] etc. A primary step for understanding sleep issues, is the diagnostic process Polysomnography (PSG) of which sleep stage classification is a crucial component [31]. It is understood that within sleep there are distinct different stages present [32, 14, 33]. These stages are identified as: Awake; Stage 1 Non-Rapid Eye Movement (NREM); Stage 2 NREM; Stage 3 NREM; Stage 4 NREM; and REM (Rapid Eye Movement) [32, 14, 34]. For simplicity, stages 1 and 2 can be unified as light sleep [35], and stages 3 and 4 are often combined as Slow Wave Sleep (SWS) or deep NREM sleep [32, 14]. For each sleeper these stages occur semi-periodically during the course of sleep, however not all stages are equally spaced in time and the sleeper does not go monotonically through all stages [14]. It is also known that certain sleep stages dominate depending on how long the person has slept [14]. In addition, each sleep stage is associated with a unique physical or cognitive purpose. For example, REM sleep is linked to memory consolidation and learning. Whereas during deep sleep bodily cell repair takes place [36].

The gold standard for identifying the stages of sleep is the use of brain information collected non-invasively from the electroencephalogram (EEG) by placing small electrodes on the head [32, 14]. In addition to brain activity, other physiological changes can be noted specific to each stage of sleep, including: eye movement; body movement; and muscle activity. For example, comparing NREM stages to being awake: heart rate (HR), blood pressure, breathing rate (BR), core temperature, and responsiveness to CO_2 all decrease [14, 34, 37]. Between REM and NREM, an increase in amplitude and variation for HR is noted, and the BR increases and fluctuates more [14, 34, 37, 38]. In addition, NREM sleep

stages are associated with core body cooling, this in turn corresponds to increasing skin temperature the body radiates heat away from the core [39].

Thus for monitoring sleep and diagnosing sleep disorders PSG is used, where an array of sensors are applied to the user in order to study the parameters discussed above [40, 41]. The sleep stage is then determined by a process of scoring based upon the sensor data, and changes in the sleep architecture throughout the night can be used to help diagnose and monitor different sleep-related conditions. The commonly used Rechtschaffen and Kales (R&K) and American Association of Sleep Medicine (AASM) guidelines require visual analysis which is extremely time consuming [42]. Hence, sleep analysis performed via automatic classification methods is gaining increasing interest, with results similar to those of an experienced PSG technician having been demonstrated [42, 43].

1.3 Sensors in Extreme Environments Background

Regarding extreme environments, sensors must be designed and constructed to withstand the harsh conditions and provide accurate measurements. This is important because sensors play a critical role in many industries, and the development of sensors that can operate in extreme environments has enabled new applications and advancements in science and technology. Literature demonstrates the use of sensors in extreme environments for a range of different circumstances. Such circumstances may include: low/high pressure; temperature extremes; radiation extremes; impact extremes etc. For these examples, the sensors employed are often highly specialised and engineered specifically for a particular use case which can result in increased sensor costs. The remainder of this section shall discuss the challenges of sensing in the extreme environments highlighted and discuss some of the known approaches to mitigating these difficulties. Particular focus will be given to the specific environments investigated as part of this thesis (space and underwater).

1.3.1 Impact

In high impact situations, caused by shock or severe vibrations for example, robust sensor technology is required. Such conditions may occur in vehicles, military equipment or seismic analysis [44]; where mechanical considerations are employed to limit the damage caused. This may involve minimising the component weight to limit momentum or increasing strength of the component casing to improve rigidity [44]. In particular, accelerometers are often used in high impact environments, with certain devices being able to withstand shocks up to ±600000g [45].

1.3.2 Temperature

Temperature presents a significant challenge with regards to sensor deployment [44]. This is because of the direct relationship between temperature and electrical resistivity, detailed by figure 1.3.

As demonstrated the resistivity of given conductors increases with temperature, this directly effects the electrical resistance of these conductors as demonstrated by equation 1.1. Where R is the resistance of the conductor, ρ the resistivity of the conductor and A the cross sectional area of the conductor. Resistance is a fundamental circuit property, that defines how many electronic devices operate. This can cause significant difficulties with regards to sensors or any electronic component operating in temperature extremes, as electrical resistance dictates many circuit properties such as current draw [47] and power dissipation [47].

$$R = \frac{\rho L}{A} \tag{1.1}$$

In addition to this, PN junctions are also affected by temperature. PN junctions

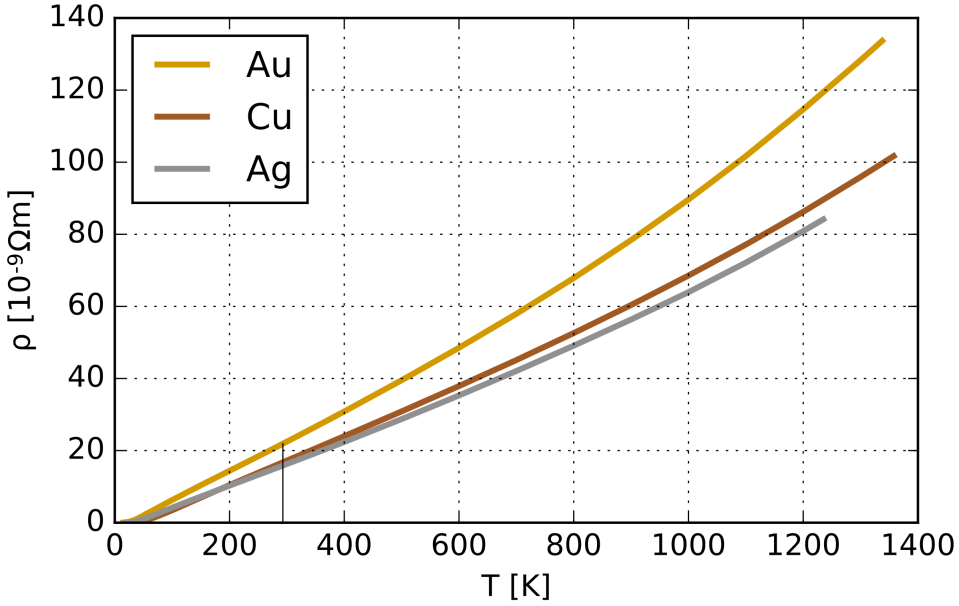


Figure 1.3: Affect on resistivity of gold (Au), copper (Cu) and silver (Ag) relative to increasing temperature. Taken from [46].

are interfaces between two types of semiconducting material that are crucial to modern electronics [48]. Equation 1.2 demonstrates the ideal diode equation, which describe the current flow through a diode (PN junction [49]. Where I is the current flow through the diode, I_0 diode leakage current, V voltage applied to diode, q charge of single electron, k Boltzman constant and T absolute temperature (K) [49].

$$I = I_0(e^{qV/kT} - 1) \tag{1.2}$$

From equation 1.2 it is clear that the current flow through a PN junction is dependent on temperature and therefore significant temperature variation can cause inconsistent semiconductor performance. The implications of this extends to transistors as both Bipolar Junction Transistors (BJTs) [50] and Metal Oxide Semiconductor Field Effect

Transistors (MOSFETs) are also comprised of PN junctions [51]. Each of these transistor variants are crucial to sensing applications. BJT transistors are often used for signal amplifying purposes and MOSFET transistors are used to form crucial modules within micro-controllers/Central Processing Units (CPUs). Generally, the maximum operating temperature for PN junctions approximately 125 °C [52, 53] and minimum is is around -40 °C [54]. Solutions to the issues discussed here include isolating the electronics from the extreme temperature via use of thermal insulation. Or, in the case of high temperature, use of water cooling. However, such solutions will increase design complexity/cost [44].

1.3.3 Pressure

Sensing in extreme pressure conditions can limit the performance and durability of sensor devices due to the strenuous conditions associated. Extreme pressures can be found in space, vacuum chambers or deep underwater. As such, certain sensors are required to operate in these conditions. For example, Piezoresistive pressure sensors are known for their ability to withstand high/low pressure conditions and are commonly used in applications such hence are widely used [44]. In addition quartz-based sensors, are specifically designed to measure and withstand high pressures, whereas Barometric pressure sensors are used to measure air pressure at or close to the Earth's surface [44].

1.3.4 Radiation

Highly radioactive environments can alter both the mechanical and electrical properties of metals and semiconductors resulting in poor sensor performance. This can occur when sensing in/around nuclear power plants or during space flight [44]. Specifically, semi conductor devices are particularly prone to radiation damage hence are avoided where possible in the nuclear industry [44]. Other types of sensors are naturally more resilient to high

radiation in particular electromechanical devices [44]. Appropriate senor selection or radiation shielding the two main solutions when attempting to mitigate the effects of radiation on electronic sensors.

1.3.5 Micro-gravity

Sensor deployment in space is affected by all of the issues discussed above. High radiation, high impact and temperature/pressure extremes are all associated with space flight hence sensor selection/fabrication must mirror these factors. Sensing is a crucial aspect of space flight, especially how the sensor data is processed and used. For example, sensor data is utilised for monitoring the space craft, e.g. the vessel hull [55]. In addition, wearable sensor data is used to monitor the humans on board the ship [56]. As stated previously, this thesis shall focus on the monitoring of humans in microgravity, specifically the analysis of sleep in space.

Astronauts experience poor sleep quality in microgravity and this is well documented in literature [57, 58, 59]. This is not ideal considering that astronauts work on/with very expensive equipment in an environment that is potentially extremely dangerous. In addition, with the recent expansion of the private spaceflight sector, it likely that more human beings will be spending longer periods in space. As such, sleep study has emerged as an important branch of space medicine which has identified issues such as circadian rhythm misalignment [57, 59]. Circadian rhythms are physical, mental or behavioural changes [60] that follow a daily cycle (e.g. sleeping at night, wakefulness during the day). These rhythms primarily respond to changes in environmental light, i.e. the 24 Hr day and night cycle on Earth. This misalignment is caused by variation in light/dark cycles and workload scheduling issues [57, 59, 61] whereby the inconsistent working hours required of astronauts results in an inability to adhere to a regular routine [57, 61]. Sleep environment is another

factor which plays a role in astronaut sleep disruption. Studies have demonstrated that the habitats in which astronauts sleep have unsuitable ambient temperatures, higher noise levels and uncomfortable sleeping arrangements [57].

The problem with utilising a full PSG configuration, describe by section 1.2.1, in a micro-gravity environment is associated with the large array of sensors and other equipment that must be deployed. This issue is highlighted by the fact that space craft are often cramped environments where weight considerations are of paramount importance [62]. Also, as discussed full PSG utilises techniques such as EEG which requires multiple electrodes attached to the user. This is not ideal considering the already uncomfortable experience of sleeping in micro-gravity.

To address these issues, the solution should be lightweight, comfortable and compact demonstrating full sleep stage identification (not just sleep onset, duration etc). As with the theme of this thesis, a wearable chest strap sensor platform was constructed. Sleep analysis is achieved via a multi-sensor analysis approach, combining acceleration, ECG, temperature changes, and breathing rate estimates. Any devised method must be bench marked against existing technology, for this work comparison was made to the sleep stage predictions from a Fitbit smart watch. This device (Fitbit Versa 2) measures: wrist acceleration; Heart Beats Per Minute (BPM) and Heart Rate Variability (HRV) via PPG The watch been demonstrated to be: 61% accurate for detecting awake; 81% accurate for detecting light sleep; 49% accurate for detecting deep sleep; and 74% accurate for detecting REM [63]. Analysis is performed against the Fitbit device considering the difficulty associated with accessing PSG equipment especially considering this work was performed during the Global pandemic. However, despite the limitations of the Fitbit device, it is believed to be one of the better performing wrist worn devices and has been assessed against PSG equipment (by Zambotti et al [63]). In addition to smartwatch technology, other

non gold standard technology is commercially available. For example the company Emfit produce sleep analysis tools, notably the QS 2 [64], which demonstrates good sleep stage classification performance. However, this device must be positioned underneath a mattress requiring good contact between bed, bed frame and sleeper in order to function well. This would not be practical for deployment into space where the lack gravity cannot ensure the contact required; instead some fastening approach would be necessary which would likely be uncomfortable. A wrist worn device would not be susceptible to such issues. However, the discussed inaccuracies associated with wrist worn sleep trackers and associated physiological monitoring methods, for example PPG, is a key motivator for the investigation of a chest worn device. The chest provides easy access for recording the ECG for motion robust heart monitoring [16, 17], the measurement of breathing rate (e.g. [65]) and ECG R peak variations. In addition, better estimation of core temperature compared to the the wrist can be achieved [66, 22].

1.3.6 Underwater

As humans, it is understood that water and electricity should not be brought into contact with each other primarily due to the risk of electrical shock. This understanding extends to electrical components which risk short circuits in the presence of water causing damage to sensitive components. Pure water is not a good conductor of electricity [67], but the presence of dissolved ions in water increases the conductivity of the solution allowing current to flow [67]. The resultant conductivity depends certain factors such as: the concentration of the dissolved ions [68] or the temperature of the solution [69]. The conductivity of an aqueous solution is often represented as micro Siemens per metre (µS/m) or milli Siemens per metre (mS/m) etc [68]. For example, freshwater has conductivity of 0.01 S/m [70] compared to seawater which typically has a conductivity of: 4 S/m [70]. To highlight the

discussed relationships, figure 1.4 demonstrates the trend between concentration of sodium chloride (NaCl) and the aqueous solution conductivity. As demonstrated the concentration of NaCl increases the conductivity. In addition, figure 1.5 demonstrates the relationship between increasing the temperature of pure water, and the resultant aqueous conductivity. As seen, increasing the temperature also increases the conductivity. A simple solution to the issues discussed, is to waterproof the electronics. This can be achieved by enclosing the components in a suitable waterproof enclosure.

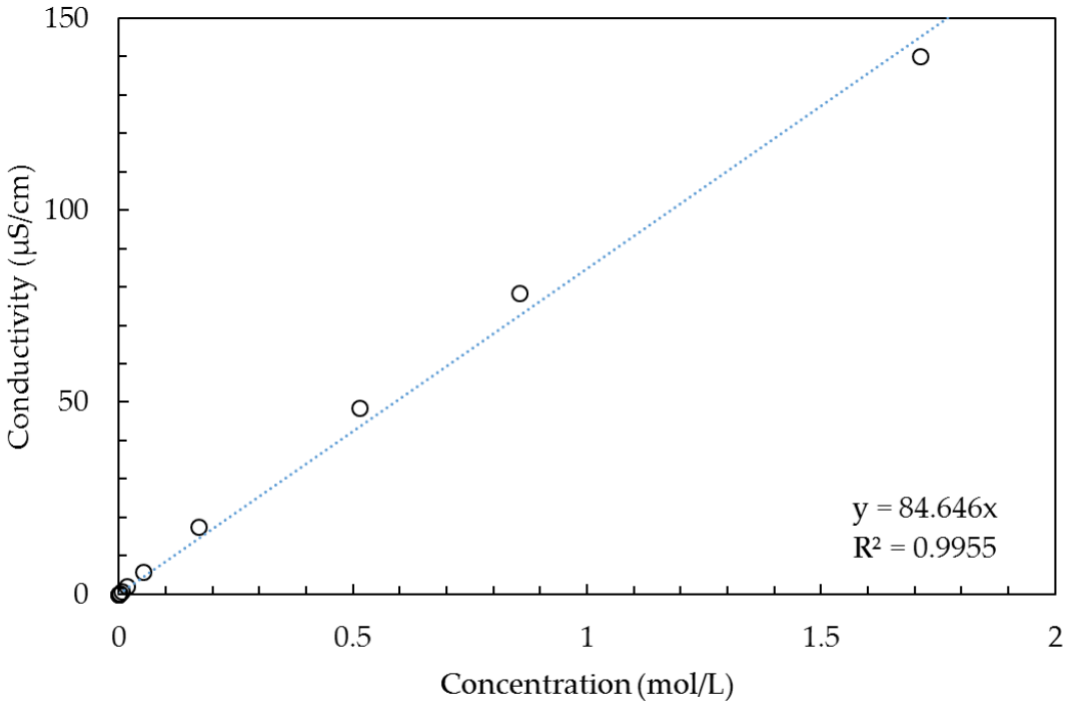


Figure 1.4: Affect on solution conductivity relative to increasing concentration of sodium chloride (NaCl). Taken from: [71].

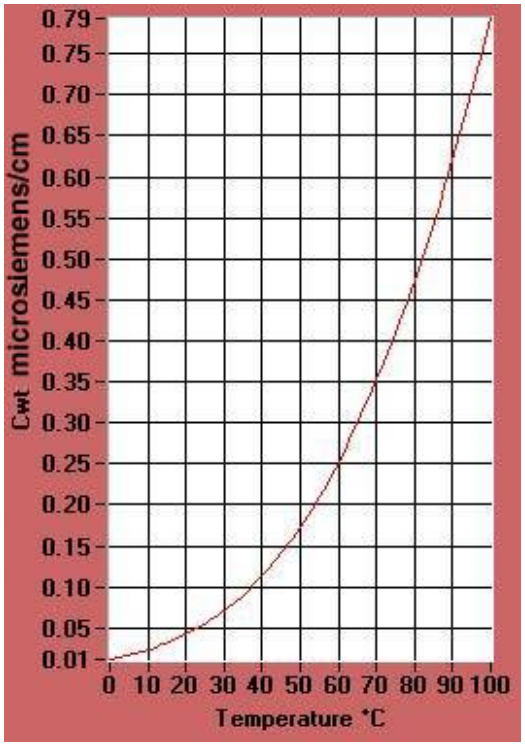


Figure 1.5: Affect on conductivity of pure water relative to increasing temperature. Taken from [72].

For the underwater applications considered in this thesis (swim analysis), the sensors used are affected by the presence of water. Specifically, the ECG sensor, which requires good contact between skin and electrodes. However, ECG is desirable considering it the gold standard approach to cardiac analysis. There are many important metrics that can be observed from ECG data in any environment, arguably the most important being BPM as it is a crucial physiological marker in human beings [73]. Hence, is the focus for underwater ECG analysis in this thesis. In the time domain, to calculate BPM from an ECG trace, one of the wave components must be identified per heart beat. Arguably the QRS complex (see figure 1.1), characterised by the sharp signal (R) peak, is the easiest feature to locate. The difficulty associated with underwater ECG analysis and therefore generation of underwater

BPM is caused by the signal noise and distortion resultant from underwater deployment which is discussed further in chapter 5.

In literature, solutions exist to this problem including electrode fabrication methods. This involves utilising sealed Ag/AgCl electrodes or carbon powder electrodes [74, 75]. Such electrodes can yield clean underwater ECG traces suitable for further analysis, however the Ag/AgCl electrodes are time consuming to apply and are reported to cause severe skin irritation due to the suction associated with fixing/sealing the electrodes [75]. Carbon powder electrodes are not associated with such difficulties and can also generate clean ECG traces especially when combined with poly-dimethylsiloxane [74]. However, this process is chemical in nature adding complexity to the design. Furthermore, Stauffer et al introduce fabrication of polymer electrodes intended for clinical applications [76], also demonstrating functionality when tested underwater. However such electrodes are complicated with regards to fabrication. An alternative solution to electrode design discussed in literature is focusing on amplifier considerations. A solution proposed by Gradl et al utilises a transimpedance amplifier circuit including active current feedback to produce clean underwater ECG waveforms [77]. Post amplification, additional filters are required to further clean the signal. This process also demonstrates some good results but increases electronic circuit complexity. Whilst all solutions discussed here (electrode/amplifier design) can achieve usable underwater ECG measurements suitable for calculating BPM, there are demonstrated design/use issues and complexities associated with each.

In addition, in literature, no software focused solutions to underwater ECG interpretation can be found. There are however demonstrated techniques for general ECG analysis of low Signal to Noise Ratio (SNR) data. For example, Vullings et al demonstrate the effective filtering of noisy ECG data via novel signal processing techniques [78]. This method is compared to existing wavelet approaches and is demonstrated to perform well. There

are also several examples of machine learning methods that are intended for application to ECG signals, many solutions can be found that classify abnormalities such as arrhythmia [79, 80]. More relevant, other machine learning techniques demonstrate R peak detection via unsupervised clustering methods [81] or using supervised convolutional neural networks [82]. The supervised neural network approaches appear to perform significantly better.

Commercially, devices exist intended for swim fitness tracking. In particular, the Garmin HRM Swim strap [83] tracks BPM from the chest by using an ECG sensor during swimming. This product is one of the best performing devices available on the market for this purpose. Garmin highlight the non-slip strap is specifically designed for use in swimming pools however will also work in open water. Emphasis is placed on the design of the strap with regards to ECG signal generation. Garmin do not indicate if any specialised signal processing is necessary to produce useable data, nor how accurate their device is with regards to generating BPM and no Garmin HRM Swim BPM accuracy information can be found in literature. Note, Garmin state in order to use the HRM chest strap, a compatible Garmin watch is also required. In addition to the strap discussed above, Garmin (and other companies) produce wrist worn devices capable of monitoring swimming heart rate via PPG. However, nominal (land) PPG has issues as per section 1.2. Underwater PPG presents further difficulties including decreased signal amplitude and SNR at lower water temperatures [84].

As well as underwater BPM calculation, swim stroke classification via chest situated accelerometer data is another factor investigated as part of the underwater experimentation. In literature, examples of classifying swim stroke via application of machine learning techniques to accelerometer data have been demonstrated. These approaches highlight stroke classification from the wrist [85], lower back [86] and legs (using three individual 9 DOF IMUs per leg) [87]. No examples of chest based swim stroke classification have been

found. Of the examples seen in literature, the wrist and the legs in particular demonstrate notable classification performance. The intention of this work however, as discussed, is to situate all components on a chest strap that is relatively consistent in design between deployment environments. In addition, to classify swim stroke from the legs, three sensors per limb are required which is likely cumbersome to prepare prior to swimming, in addition cost is increased considering the need for six devices. Commercially, waterproof wrist worn devices (smartwatches) are available that can classify swim stroke during swimming. In particular devices built by companies such as Apple (Apple Watch [88]) and Garmin (Garmin swim [89]). As with similar technology discussed in this chapter, information on the accuracy of such devices has not been made publicly available and no validation studies have been found in literature. In addition, wrist/leg situated technology may not be suitable for disabled swimmers compared to a chest situated platform.

1.3.7 Humidity

Similar to underwater sensor deployment, sensing in humid environments can also pose challenges due to the presence of moisture that causes damage to electronics devices. As before, a solution to this involves isolating components from humid conditions via use of an appropriate enclosure (water proof). Or, by using a humidity sensor to deactivate electronics in the presence of moisture to limit potential damage. For example, capacitive humidity sensors measure the change in dielectric properties of a humidity-sensitive material, typically a polymer, in response to moisture absorption [81]. Such devices are commonly used in applications such as environmental/agriculture monitoring [81].

1.4 Conclusions

This chapter introduces this thesis by detailing the key aims & objectives for this work. As discussed, the focus was to develop a sensor platform capable of sensing in different environments without substantial electronic modification using low cost off the shelf sensors. To compensate for this, emphasis was directed towards signal/data processing to account for the lack of specialised sensors. The sensors used are physiological in nature given the use cases (swim/sleep analysis), hence an overview of biomedical monitoring devices is detailed. After which a review of sensors in extreme environments is given. Attention is directed towards the environments focused on in this thesis where the literature associated is discussed prior to the subsequent results.

The remainder of this thesis is structured as such: chapter 2 shall introduce the signal/data processing methods used in this work; chapter 3 details sensor platform construction and introduces how the physiological metrics are generated (BPM, breathing rate etc); chapter 4 investigates potential data extraction issues particularly during underwater investigations; chapter 7 demonstrates the initial results from the sleep investigation; chapter 5 focuses on the generation of underwater BPM; chapter 6 considers all sensors during underwater use to analyse swimming generally and chapter 8 investigates if the techniques investigated in chapter 6 are applicable to sleep data to improve sleep stage predictions. Finally, conclusions are given in chapter 9.

Chapter 2

Signal/Data Processing Techniques

Modern computing technology is associated with ever increasing computational capabilities, meaning that sophisticated computer algorithms can be designed and deployed on low-cost computers. Data processing, particularly machine learning, sits at the forefront of research within Computer Science meaning new techniques/methods frequently become available. These two considerations highlight the reasoning for focus being directed towards a data processing approach opposed to physical sensor design with regards to deployment into extreme environments. This chapter shall discuss the data processing techniques used as part of this work providing an overview of each method with regards to functionality.

Analysis techniques are either applied to the raw sensor data (e.g. ECG data) or processed sensor data (e.g. BPM from ECG) dependent on the use case. Different processing methods may be required depending on the circumstance considered, this is highlighted by focusing on BPM generation. For certain cases in this work, BPM may be determined via a peak analysis algorithm considered to be relatively simple. However, in other situations such a method may not be suitable given increased noise present on the ECG signal for example. This may render peak detection error prone. In such circumstances, more

complex algorithms for example a neural network may be necessary to resolve the noise as detailed in subsequent chapters. Therefore, it is crucial to explain all techniques used within this work so that a clear understanding of all methods is granted. Many of the applied techniques can be categorised into the following: machine learning; control theory or signal processing. These methods are implemented using either MATLAB or python (specific libraries e.g. Pandas, Sci-kit learn, Tensorflow, Keras etc) both of which implement the processing methods required. These software packages are commonly used in the fields of Engineering & Mathematics considering the large range of inbuilt functions that are mostly pre programmed and require minimal modification to enable basic functionality. In addition, many of the python packages are open source meaning that instances where in depth configuration was required, assess to the base code is granted. Where appropriate, subsequent chapters shall reference back to this chapter so that any signal/data processing algorithm used in this work can revised here.

2.1 Signal Processing Algorithms

2.1.1 Peak Detection

The peak detection function (findpeaks) allows for the identifying local signal maxima [90]. MATLAB state this function identifies a peak if a data point is larger than its two neighbouring samples. Control of this algorithm is granted via two main parameters: minimum peak height (value required to trigger peak identification) and peak spacing (minimum number of samples required prior to next peak identification). Configuration of these parameters allows the findpeaks function to be applied to specific use cases as demonstrated by figure 2.1. Here findpeaks is applied to a normalised ECG sample so to locate the R peaks. As demonstrated this is an effective method for this clean ECG signal

and is computationally inexpensive way of doing so.

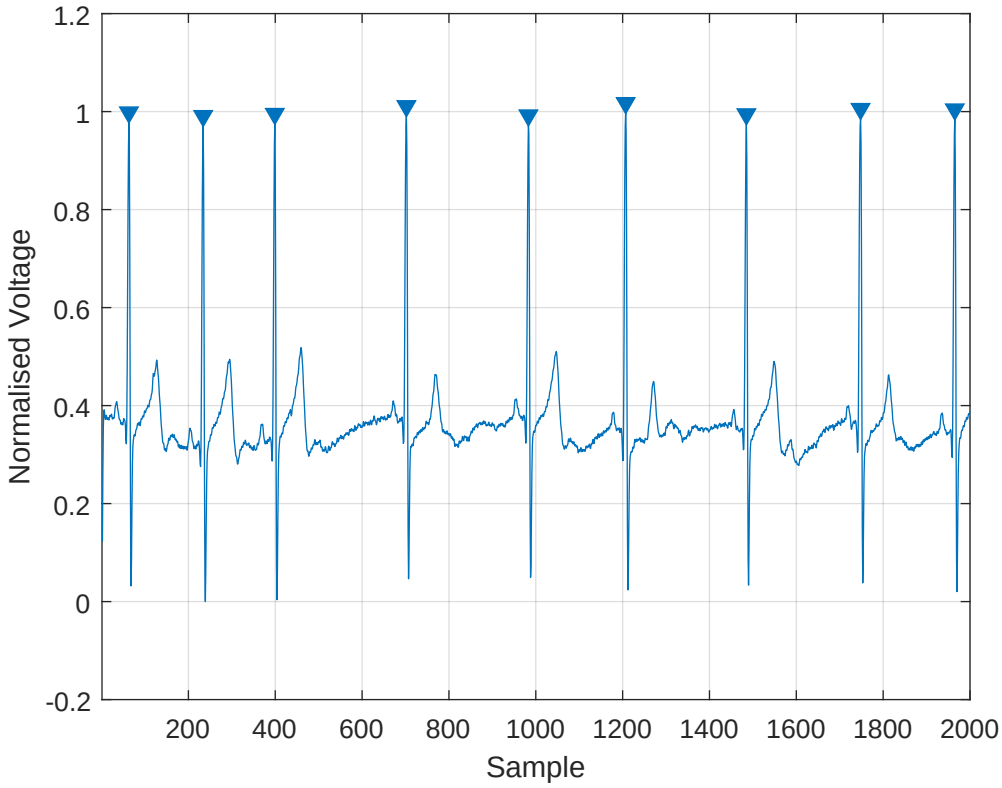


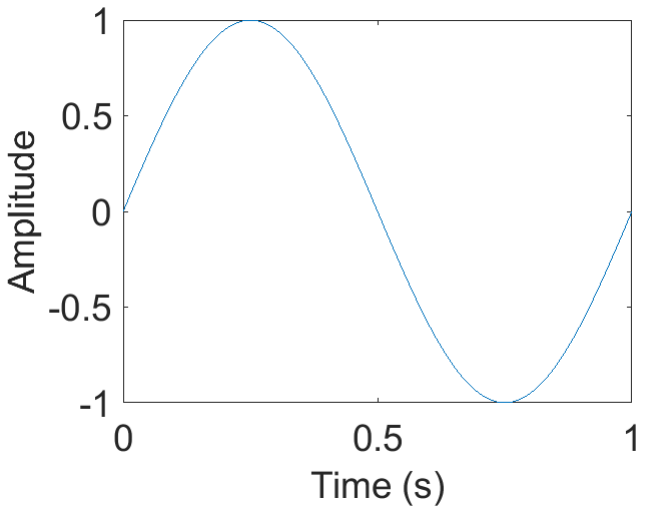
Figure 2.1: Findpeaks function applied to normalised ECG sample. As demonstrated, R peaks effectively located for this regular ECG sample.

2.1.2 Fast Fourier Transform

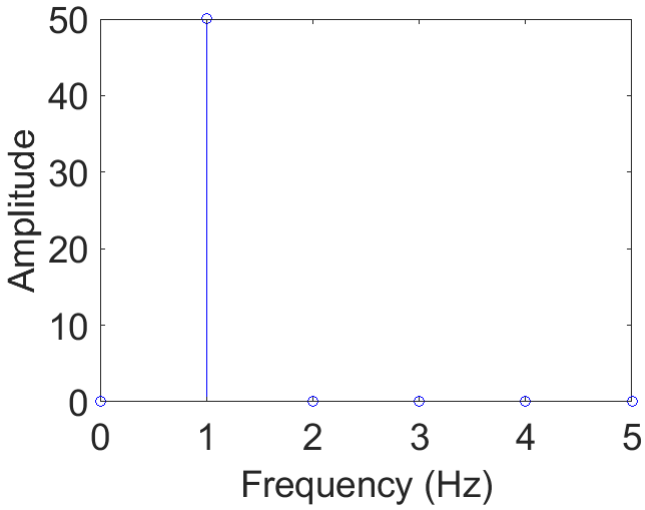
The Fast Fourier Transform (FFT) is an algorithm used to compute the Discrete Fourier Transform (DFT) of a signal in computationally undemanding manner [91]. The DFT is used to represent a discrete signal (sampled in time) in the frequency domain [91]. This allows for identification of the key frequencies which is particularly useful when analysing noisy/multi component signals. The FFT has many applications, within this thesis it

is applied generally when assessing sinusoidal signals. For example, the estimation of breathing rate by identifying the dominant frequency component of a sinusoid, generated by a stretch sensor orientated around a persons chest. Given the rhythmic nature of such a signal, it is the optimal choice of algorithm for this purpose. The full mathematics of this process are not discussed in this thesis however can be found at [92]. Implementation of the FFT algorithm is performed usings MATLAB's FFT and FFTshift functionality [93]. Figure 2.2 demonstrates application of FFT in MATLAB where a 1 Hz sin wave (figure 2.21) is represented in the frequency domain (figure 2.22). As demonstrated, a clear peak is visible at the 1 Hz due to the appropriate configuration of FFT in this case. Here the signal is sampled at 100Hz and the number of points used by the FFT algorithm (NFFT) is 100. Therefore the frequency resolution is 1 Hz as per equation 2.1 where Δf is the frequency resolution and f_s the sampling frequency [94]. Given the frequency resolution is 1 Hz, and the frequency of the signal is 1 Hz, the appropriate results are realised.

$$\Delta f = \frac{f_s}{NFFT} \tag{2.1}$$



(1)



(2)

Figure 2.2: 1 Hz sin wave shown by 2.21, represented in the frequency domain by 2.22 via use of FFT. As shown, clear peak visible at 1 Hz.

This however may not always be the case, it is often optimal to configure NFFT as a power of two so to minimise computational cost. If, for the case discussed above, NNFT

were configured as 128 the frequency resolution would be 0.78 Hz and therefore 1 Hz (the frequency of the signal) could no longer be resolved as demonstrated by figure 2.3. As shown here, the FFT does not identify 1 Hz as the sole frequency component of this sin wave, energy has 'leaked' into adjacent frequencies. This is referred to as spectral leaking and is undesirable as it limits the ability of the FFT algorithm to appropriately locate the frequency components of a signal.

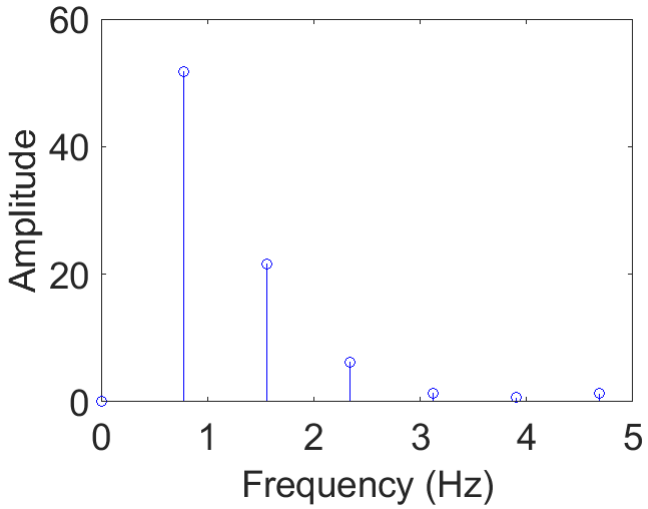


Figure 2.3: FFT applied to 1 Hz sin wave (same as figure 2.21) however spectral leakage has occurred due to frequency resolution.

To limit spectral leakage, a windowing function can be applied. This tapers the signal, reducing the abrupt changes hence minimising spectral leakage. For this work, a rectangular window is used for this purpose implemented via MATLAB [93].

2.1.3 Continuous Wavelet Transform (CWT)

CWT is a signal processing tool that analyses signals in time–frequency space and reduces noise whilst retaining the important signal characteristics, it is particularly useful for analyzing non-stationary signals [95]. Relative to this thesis, this factor means CWT is a

viable technique for the identification of R peaks from noisy ECG signals, as per figure 2.4. As mentioned before peak analysis is a relatively simple technique, therefore it is computationally inexpensive however not always suitable especially for noisy signals. Hence, CWT can be considered. Crucially, CWT involves the convolution of an input signal with a selectable function. The convolution operation measures the similarity between the wavelet and the signal at each point in time, producing a two-dimensional representation called the scalogram or CWT coefficients. CWT coefficients represent the strength or energy of each wavelet at different time points and scales. The scale parameter controls the width of the wavelet, determining the frequency resolution, while the translation parameter determines the position in time where the wavelet is centered. By varying these parameters, CWT can analyze different frequency components of the signal at different time points. The resulting scalogram provides a visual representation of how the signal's frequency content evolves over time. High-energy coefficients in the scalogram correspond to regions of the signal that match the characteristics of the wavelet at specific scales and positions. These regions indicate the presence of particular frequencies or patterns in the signal. By convolving the input signal with a family of wavelets, CWT provides a detailed representation of how the signal's frequency content evolves over time [95].

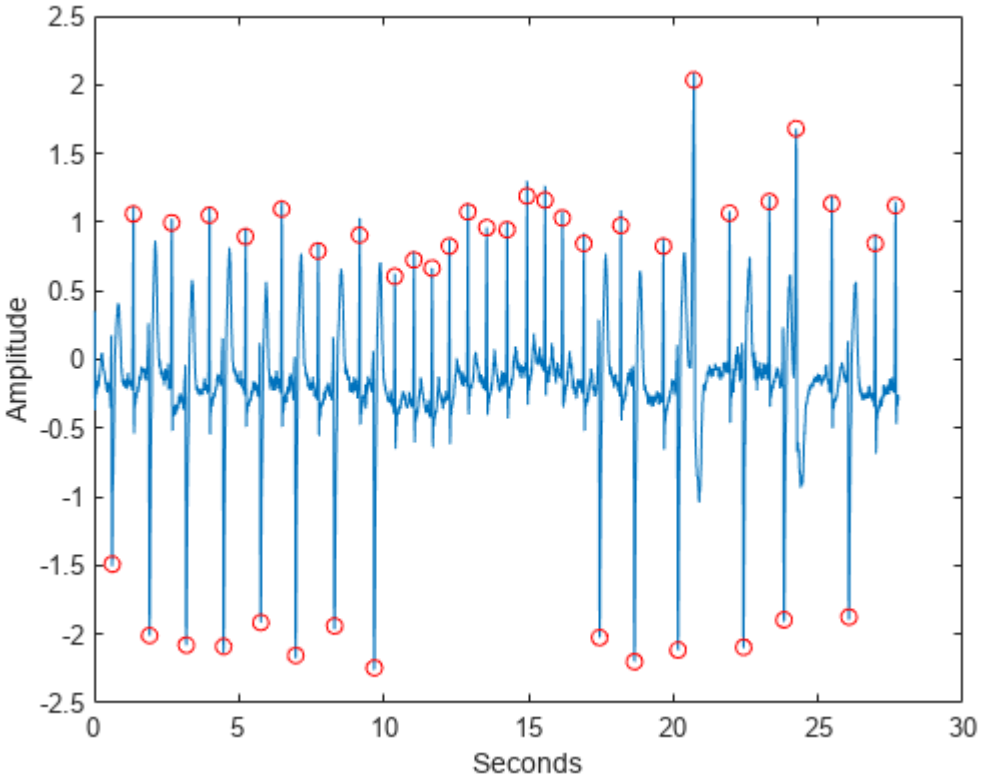


Figure 2.4: CWT applied to ECG sample. As demonstrated, R peaks effectively located from this irregular ECG sample. Taken from [96].

2.1.4 Pan Tompkins Algorithm

The Pan-Tompkins algorithm is a signal processing technique used to identify the QRS wave forms on ECG signals. Hence, unlike the previously discussed signal processing algorithms, it is implemented specifically for one task: R peak detection [97]. The algorithm involves multiple steps detailed below.

Filtering: The ECG signal is initially filtered using a bandpass filter to remove noise and unwanted frequency components. Typically, a combination of low-pass and high-pass filters is used to enhance the QRS complex while attenuating baseline wander

and high-frequency noise. Differentiation: the filtered signal is then differentiated to emphasize the slope changes in the QRS complex. This step amplifies the high-frequency components associated with the steep slopes of the QRS complex. Squaring: the differentiated signal is squared to emphasize the amplitude of the QRS complex and suppress the polarity changes. This process also helps in maintaining the waveform's positive polarity. Moving Window Integration: a moving window integration technique is applied to smooth the squared signal. The integration process calculates the sum of squared amplitudes within a specific window size. This window is shifted over the signal to obtain a series of integrated values. Adaptive Thresholding: an adaptive thresholding technique is used to determine the appropriate threshold for QRS complex detection. The threshold is set dynamically based on the characteristics of the integrated signal, such as the average amplitude and the standard deviation. This distinguishes the QRS complexes from the background noise. Peak Detection: with the results generated from adaptive thresholding, peak detection is applied to the integrated signal. Identified peaks are the R peaks.

These stages combined makes the Pan-Tompkins algorithm an effective tool for identifying R peaks on ECG signals. It is widely used in clinical practice and ECG analysis software to extract important features for diagnosing various cardiac conditions [97]. For this work, the algorithm is implemented via the Pan Tompkins algorithm toolkit available within MATLAB [97].

2.2 Machine Learning Algorithms

Machine learning is the field of study that gives computers the ability to learn from data without being explicitly programmed to do so [98]. To construct a machine learning algo-

rithm, it is crucial to gather some training data relative to a specific task for the program to learn from. The task can be any realistic application including spam filtering [99]; predicting house prices [100]; film/music suggestions [101] etc. The performance of the algorithm is measured by some metric such as accuracy (ratio of correct predictions [99]) however selection of this metric is dependent on the application [99] and will be highlighted in subsequent sections. Machine learning is highly suited to tasks which exhibit the following:

- Existing solutions require a lot of fine tuning or long lists of rules [99]
- Complex problems for which a traditional approach yields no good solution [99]
- Fluctuating environments where new data is different to previous data [99]

For this thesis, the machine learning algorithms featured can be categorised either as supervised or unsupervised. The crucial difference between these two is that supervised machine learning algorithms require training data to be accompanied with associated solutions (labels) [102]. Typical supervised problems include classification (predicting a category based on data) or regression (predicting a value based on data) [99]. Fundamentally, classification outputs the probabilities associated with a given data instance belonging to a particular class. The class indicating the highest probability is then identified as the classification for this data instance [103]. Regression, instead of outputting a probability, outputs a continuous number based on the input [99]. Classification can be sub divided into two sub categories: binary classification or multi-class classification. Binary classification is whereby an algorithm may categorise a data instance as one of two classes [102]. Multi-class classification is whereby each data instance must be classified as one of three or more classes [102]. This thesis investigates both regression and multi-class classification type algorithms.

To assess the performance of a classification model (binary or multi-class), the simplest measure is accuracy. As stated above it is the ratio of correct predictions made by a model. However, when evaluating a class imbalanced dataset (dataset featuring notably more instances of a particular class) accuracy is not a good performance indicator. This is because according to the accuracy metric, a model may perform relatively well by predicting all data instances as the majority class. However, the performance of a model may, in reality, be defined by its ability to identify the minority class. In such cases other metrics such as precision or recall may be more useful. To interpret these alternate metrics it is first important to understand True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN). These quantities highlight a classifiers performance by specifically investigating individual classifications of the model and are used to calculate precision and recall; which are demonstrated by equations 2.3 and 2.2 respectively. Effectively, the precision identifies the accuracy of positive predictions. Recall relates to the percentage of class labels that were correctly classified [104]. As an effort to combine the recall and precision, calculating the harmonic mean of the metrics yields F1-score.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.3}$$

$$\text{F1 Score} = \frac{2(\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \tag{2.4}$$

In addition to appropriate metric selection, oversampling techniques such as Synthetic Minority Over-sampling Technique (SMOTE) [105] can be used to alleviate issues that

arise from imbalanced datasets. This method involves generating synthetic instances based on existing data to artificially increase representation of the minority class/classes. This improves performance by increasing a models exposure to all classes and increases the size of the training set which is beneficial for reasons discussed below.

To asses the performance of a regression model, the Mean Absolute Error (MAE) can be used. This metric is described by equation 2.5. Where n is the number of samples, x_i the true value and $\hat{x}_i$ the predicted value. The Root Mean Square Error (RMSE) is also used to assess model performance, this metric is detailed by equation 2.7. The RMSE is similar to the Mean Square Error (MSE), a metric which significantly punishes large errors due to the squaring of the error residual (difference between predicted value and true value) [99] as demonstrated by eqaution 2.6. By square rooting the MSE, the RMSE has the same units as the predictions/labels [99]. In addition to these metrics, the Mean Error (ME) can also be calculated to assess whether, on average and by what magnitude, a model is over or under-predicting a quantity by assessing the sign of the result (positive or negative). The equation for ME is demonstrated by 2.8. MAE/MSE/RMSE/ME values closer to 0 indicate better performance.

$$\text{MAE} = \frac{\sum_{i=1}^{n} |x_\text{i} - \hat{x}_\text{i}|}{n} \tag{2.5}$$

$$\text{MSE} = \frac{\sum_{i=1}^{n} (x_\text{i} - \hat{x}_\text{i})^2}{n} \tag{2.6}$$

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (x_\text{i} - \hat{x}_\text{i})^2}{n}} \tag{2.7}$$

$$\text{Mean error} = \frac{\sum_{i=1}^{n} (x_\text{i} - \hat{x}_\text{i})}{n} \tag{2.8}$$

In addition to these metrics, R^2 is another useful method for analysis of regression models. It is a measure that indicates the 'goodness' of fit for a given model by demonstrating how well predictions made by a regressor relate to the actual labels. To calculate R^2, the Sum of Square residuals (RSS - see equation 2.9) and the Total Sum of Squares (TSS — see equation 2.10) are used as per equation 2.11. R^2 values closer to 1 indicate better performance.

$$RSS = \sum_{i=1}^{n} (X_i - \hat{x_i})^2 \tag{2.9}$$

$$TSS = \sum_{i=1}^{n} (x_i - \bar{x})^2 \tag{2.10}$$

$$R^2 = 1 - \frac{RSS}{TSS} \tag{2.11}$$

Regarding unsupervised machine learning algorithms, the training data is not accompanied with associated solutions [99] and therefor an error function as discussed above is not applicable. Instead, such algorithms might attempt to optimise certain quantities relative to the input data so to yield the best result. Typical unsupervised problems include clustering (grouping like data instances) and visualisation (representing higher dimensionality data sets in 2D/3D spaces) [99].

Solutions to unsupervised/supervised machine learning problems require an appropriate model to be selected depending on the task (classification for example). Certain machine learning techniques have more model parameters available for manipulation during the training process which are used to direct an approach towards the/an optimal solution. Model parameters are associated with increased complexity. I.e the more parameters a model has the more complex it is.

Depending on the model selected/designed, a learning algorithm such as gradient descent may be required [99]. Gradient descent is an optimization algorithm that iteratively tweaks model parameters (as discussed above, associated with model complexity) to minimise the error between the true output and the predicted output. The error is represented by a cost function (MSE for example), which is the penalty associated with a models incorrect predictions relative to the model parameters [106]. Gradient descent attempts to locate the minimum of the cost function so to yield best model performance. This is done by calculating the gradient of the cost function relative to a model's parameter vector as demonstrated by equation 2.12 where $MSE(\boldsymbol{\theta})$ is the MSE of a model prediction relative to a parameter vector, $\boldsymbol{\theta}$, of θ_0 to θ_n model parameters. ∇_θ is the vector gradient of this MSE cost function [106].

$$\nabla_{\boldsymbol{\theta}} MSE(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial}{\partial \theta_0} MSE(\boldsymbol{\theta}) \\ \frac{\partial}{\partial \theta_1} MSE(\boldsymbol{\theta}) \\ \vdots \\ \frac{\partial}{\partial \theta_n} MSE(\boldsymbol{\theta}) \end{pmatrix} \tag{2.12}$$

The gradient indicates the appropriate 'direction' to alter the parameters to minimise the cost function. Note, the vector gradient indicates direction of steepest gradient so alterations to the parameter vector must be in the opposite direction (negative) to the calculated gradient. This concept is demonstrated by equation 2.13 where $\boldsymbol{\theta_n}$ is the updated parameter vector and $\boldsymbol{\theta_{n-1}}$ the parameter vector prior gradient descent step [106]. η is a parameter called the learning rate and is discussed below.

$$\boldsymbol{\theta_n} = \boldsymbol{\theta_{n-1}} - \eta \nabla_{\boldsymbol{\theta}} MSE(\boldsymbol{\theta_{n-1}}) \tag{2.13}$$

There are several important variables that control the gradient descent algorithm including learning rate and batch size. The learning rate controls by what magnitude gradient descent alters the model parameters. A high learning rate can allow a model to train faster however may cause the model to diverge from the optimal solution. A small learning rate can result in a model that will take a long time to converge to the optimal solution [99]. The batch size parameter controls how many data samples, at a time are passed to the gradient descent algorithm. For example, if a dataset has 320 samples and batch size is set to 32, the gradient descent algorithm will individually process 10 batches of size 32 samples. Once all batches have been processed by the gradient descent algorithm, 1 epoch has passed [99]. A model may run for hundreds/thousands of epochs to converge. For details regarding the full mathematical derivation of the gradient descent algorithm, please refer to [99]. Gradient descent can be subdivided into three sub categories: stochastic gradient descent (batch size set to 1), batch gradient descent (batch size set to size of training dataset) and mini batch gradient descent (batch size set to between size of training dataset and 1). Once training is complete, the model can then be used to make predictions on unseen data. A model is said to generalise if it performs well on unseen data. If a model does not generalise well this may be because of a range of reasons. Specifically, there may be data or model configuration issues that must be addressed.

Data Issues

- Non - representative training data - if the training data does not represent the unseen data to which the model will be applied the resultant model will not generalise well [107].
- Erroneous data - if the training data features clear outliers/errors or severe noise the resultant model will not generalise well [107].

- Not enough training data - the size of the training data set can have a significant impact on the model performance. Banko and Brill demonstrated that with enough training data even relatively simple machine learning algorithms can perform well on complex tasks [108].

Model Issues

- Model tuning - machine learning models have hyperparameters (eg learning rate, batch size, number of epochs etc) that can be fine-tuned to improve performance. This tuning is performed prior to model training during which the hyperparameters are fixed (unlike the model parameters which are manipulated during training, by gradient descent for example) [99]. To ensure the best model performance these hyperparameters must be tuned appropriately.

- Overfitting - machine learning models may become too sensitive to the noise within a training data set resulting in the model not being able to generalise to unseen data. This issue can be mitigated by selecting/constructing a less complex model (fewer model parameters) or by regularising the model. Regularisation limits the degrees of freedom a model has over the model parameters which can minimise overfitting. Increasing the amount of training data can also prevent overfitting [107]. Methods of regularisation are dependent on the model selected and shall be discussed where relevant in subsequent sections.

- Underfitting to training data - machine learning models may not be sophisticated enough to interpret the patterns within a data set. This issue, as the name suggests, is the mirror issue of overfitting. To mitigate this a more more complex model must be used or reduce regularisation [99].

A crucial concept associated with the machine learning is the bias vs variance trade off, this relationship compares two types of error associated with machine learning algorithms [99]. Bias is where a model generates incorrect predictions due to wrong assumptions, this is likely caused by underfitting [99]. Variance indicates a models sensitivity to small to small variations in the training data and is particularly prevalent for models with high degrees of freedom which overfit [99]. Reducing a models bias increases a models variance and vice versa. To mitigate the high variance associated with low bias models, ensemble methods can be used. This involves training multiple machine learning models and aggregating the predictions to generate a holistic estimation. This often significantly improves the performance compared to using a single model, especially when aggregation methods are appropriately selected. Such methods shall be discussed where relevant in subsequent chapters.

Prior to application of any machine learning algorithm, it is important to first split a data set into three subsets: training set (uses of this set have already been discussed above); validation set and test set. As highlighted, the training set will be used as the training data by the selected machine learning model. The validation dataset is used to provide an evaluation of a model being trained on some training data. This allows for tuning, where appropriate, of a particular models hyperparameters to improve performance. The test dataset is used to provide an evaluation of the final model. Prior to training, it may be necessary to scale the data subsets appropriately. In general, machine learning algorithms do not perform well if the input data has notably different scales. To mitigate this min-max scaling (normalisation) or Z-norm scaling (standardisation) can be applied. Min-max involves shifting and scaling data to fit between a specific range [99]. Commonly, but not exclusively, between 0 and 1. This approach is demonstrated mathematically by equation 2.14 where: X_{MinMax} is the new value; a the new upper limit value; b the new lower limit

value; X current value; X_{Max} the maximum value of the dataset and X_{Min} is the minimum value of the dataset. Z-norm scaling involves subtracting the mean from the data set and dividing by the standard deviation which results in a new set with unit variance [99]. Mathematically this is detailed by equation 2.15 where: $X_{\text{Z-norm}}$ the new value; X the current value; μ the mean of the data set and σ the standard deviation of the data set.

$$X_{\text{MinMax}} = \frac{(b-a)(X - X_{\text{Min}})}{X_{\text{Max}} - X_{\text{Min}}} + a \tag{2.14}$$

$$X_{\text{Z-norm}} = \frac{X - \mu}{\sigma} \tag{2.15}$$

An alternative to splitting the dataset into three subsets (train/validation/test) as discussed above, is to use cross validation methods. In particular K fold cross validation which splits a dataset into K subsets, referred to as folds [99]. Cross validation then sequentially evaluates a given model on each fold, that has been trained on the remaining folds. For example, if K is set as 10, the selected model is trained at total of 10 times taking training data from 9 folds and evaluating against the remainder (1 fold). The results of which are analysed by calculating the mean score (MSE for example in case of a regression problem) and the standard deviation of scores. This provides an indication for the generalisation capabilities of a given model, in particular the standard deviation of scores for which a minimal value is desirable. A high standard deviation of scores indicates model predictions are not consistent across all folds of the data which highlights performance is significantly affected by changes in the training data. Naturally, the standard deviation should be considered in conjunction with the mean score, both of which should be minimised [99]. The issue associated with K fold cross validation is the requirement for multiple models to be trained, which may be time consuming. Variations of K fold cross validation used in this

thesis include repeated K fold and repeated stratified K fold. Repeated K fold repeats the training process per fold a specified number of times increasing the robustness of the evaluation. Stratified K fold is useful in classification situations as it preserves the percentage of each class from the original dataset within each fold.

2.3 Classic Machine Learning Algorithms

2.3.1 Unsupervised algorithms

Principal Component Analysis

Principal Component Analysis (PCA) is a widely used dimensionality reduction technique used in fields such as: image compression and data visualisation. Dimensionality reduction is the process of projecting a higher dimensionality data set onto a lower dimensionality coordinate system. Crucially, PCA works to increase the interoperability of a dataset whilst attempting to minimise information loss [109]. In order to achieve this, a linear orthogonal transformation is applied and the transformation projects the data onto lower dimensions called Principal Components (PCs). To minimise information loss, each PC is selected to maxise the variance, this can increase linear seperability of data improving the subsequent application of clustering algorithms. The number of PCs used during PCA is a hyperparameter which ultimately controls how many dimensions the resultant transform will be projected onto and therefore how much of the original information will be preserved. Table 2.1 details an example data set consisting of 30 data features, and 400 entries. For simplicity only 2 are shown here. Naturally, it is difficult to visualise this high dimensionality data as it would require 30 dimensions. In addition, this dataset may suffer from 'the curse of dimensionality', which is an issue affecting high dimensionality datasets that impedes optimisation algorithms from converging towards the best solution [110]. To

mitigate both of these issues PCA can be applied to this dataset, as demonstrated by 2.5. For this example, the number of components has been set as 2, which is a common choice as it allows visualisation on a 2D graph.

Table 2.1: Multi feature dataset

Feature 1	Feature 2	...	Feature 28	Feature 29	Feature 30
17.99	10.38	...	0.2654	0.4601	0.11890
20.57	17.77	...	0.1860	0.2750	0.08902

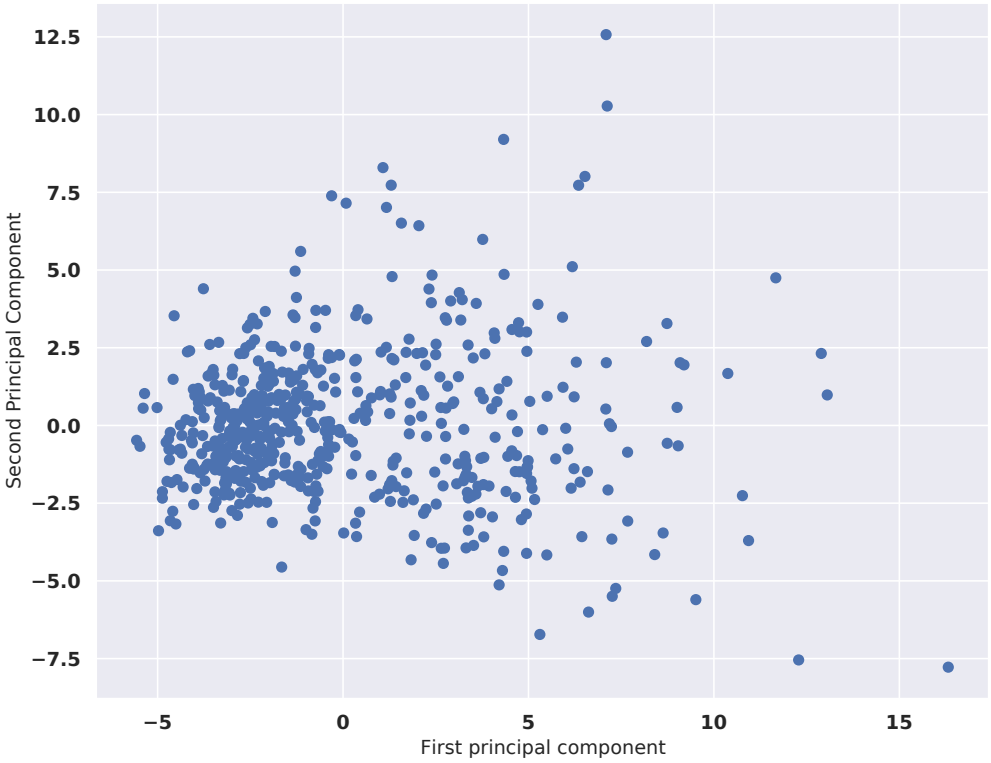


Figure 2.5: PCA (number of components set as 2) applied to dataset (Originally 30 dimensions).

KMeans

KMeans clustering is an iterative algorithm that attempts to highlight clusters within a data set. Prior to the application of this algorithm the number of clusters, K, must be specified. K centroids are then randomly inserted into the data set and each data point is assigned to the cluster of the closest centroid. The Euclidean distance from each data point to its corresponding centroid is then calculated. The mean of these distances associated with each centroid is calculated and becomes the location of the new centroids. This process is repeated until there is no change in the centroid locations [111]. PCA and KMeans are commonly used together as PCA can increase the linear separability of data which improve KMeans performance. To demonstrate this process, figure 2.6 demonstrates KMeans (number of clusters configured as three) applied to a scatter plot with three clearly identifiable clusters. The clusters are colour coded according to the KMeans results. Clearly, for this relatively trivial example, KMeans is cable of identifying these clusters.

Figure 2.6: KMeans (K=3) applied to dataset. Colour coding used to indicate KMeans results.

2.3.2 Supervised algorithms

Linear Discriminant Analysis (LDA)

LDA is somewhat similar to PCA however uses class labels to enhance separation. Where PCA devises a new coordinate system in order to maximise variance, which as a by product may improve separability, LDA creates new axes solely to improve class separability. LDA can predict the probability of a new instance belonging to a class using a combination of

the new axes and Bayes rule [112]. Note, LDA by definition assumes all data is linearly distributed.

K-Nearest Neighbours (KNN)

KNN is a rudimentary machine learning model that operates by simply storing the entire dataset. There is no training process, instead predictions are generated by inserting a new instance into the stored set and investigating the K (hyperparameter) nearest data instances. Despite this simplicity, KNN can be used for both classification and regression tasks via minor adaptation to the estimation process. For classification, the mode of the K nearest data points is calculated and this average is predicted. For regression, the mean of the K nearest data points is calculated and this average is predicted.

Support Vector Machines (SVM)

SVMs are versatile machine learning algorithms capable of both linear and non linear classification/regression tasks. SVM models are versatile however have been demonstrated to perform particularly well on classification tasks involving small to medium sized datasets [99]. The key concept behind an SVM is the separation of data by one or more hyper planes. This is demonstrated by figure 2.7 where data is separated according a particular kernel strategy, the boundaries of these planes encompass a classification 'region' whereby all instances within are classified accordingly. Training involves manipulating the the hyperplane, so that the margin that separates classes is maximised, data instances that touch the subsequent margins are referred to as the 'support vectors'. Note, the data demonstrated is from the Iris dataset (relating petal and sepal length to the type of Iris flower) [113], data instances are colour coded according to class labels. Choice of kernel allows non linear classification, use of the Radial Basis Function (RBF) or polynomial

kernel for example. However, this increases model complexity and the degrees of freedom hence risks overfitting.

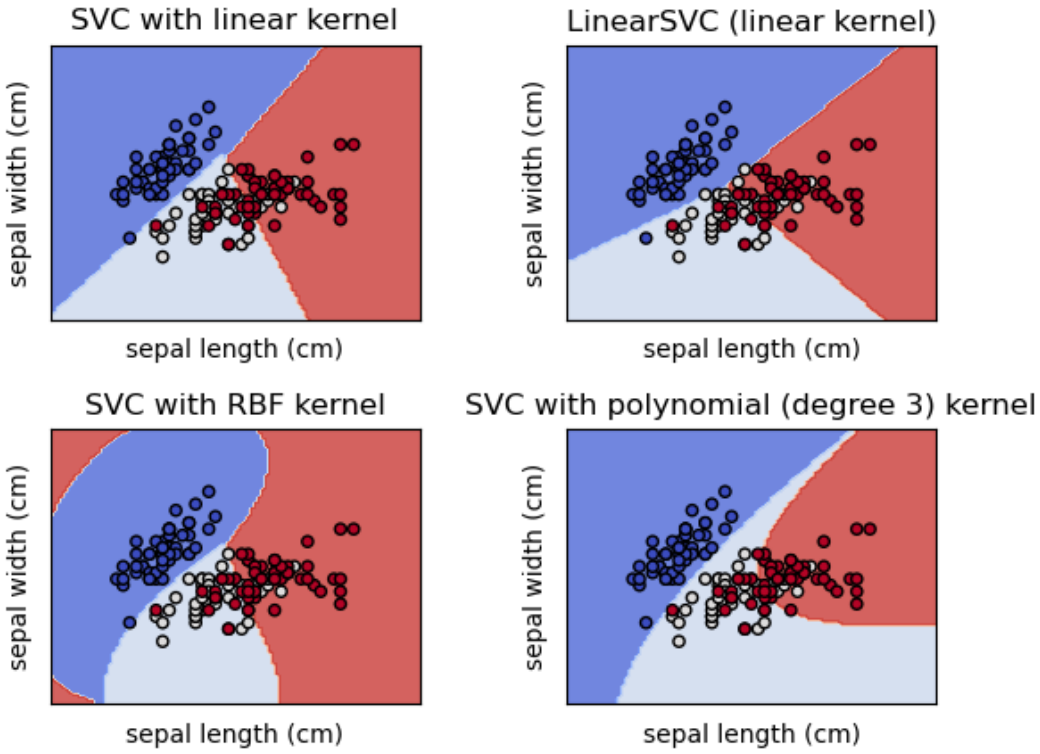


Figure 2.7: SVM applied to dataset using different kernels to classify data. Taken from: [114].

Random Forests

A Random Forest is an ensemble method involving multiple decision trees. Decision trees alone are powerful machine learning tools capable of regression and classification involving complex datasets. Training labels are used to 'grow a tree' that consists of nodes that make logical decisions based on data features, in attempt to appropriately classify/predict

data instances. An important parameter of each node is the 'gini' which indicates to how much a particular decision relates to data from each class. The gini value of a node is 0 (node is pure) if the decision only affects data instances from one particular class. Gini is an important parameter used by the Classification and Regression Tree (CART) algorithm used to train decision trees [99].

Decision trees are susceptible to overfitting, without regularisation (such as limiting the max depth of a tree) a tree can adapt itself to perfectly match the training data and thus not generalise well. A strategy to alleviate this is to limit the number of nodes the tree can create. Following on from this, small variations in the training data can result in high variance in testing results. Hence, the use of an ensemble is beneficial to assist the bias vs variance tradeoff. An ensemble of decision trees is referred to as a Random Forest the use of which often demonstrates exceptional performance. Random Forests are amongst the most powerful machine learning strategies available for deployment [99].

2.4 Artificial Neural Networks

Artificial Neural Networks (ANNs) are machine learning models inspired by the networks of biological neurons found within the human brain [99]. ANNs are extremely powerful tools that can solve highly complex problems [99]. In particular ANNs can approximate/interpret any linear or non-linear continuous function [115]. An ANN is constructed from artificial neurons an example of which is demonstrated by figure 2.8. Shown here is a neuron has 4 inputs, each of which has an associated weight which multiplies the corresponding input value (Xn). These products are then summed with an additional bias term the result of which is passed to an activation function.

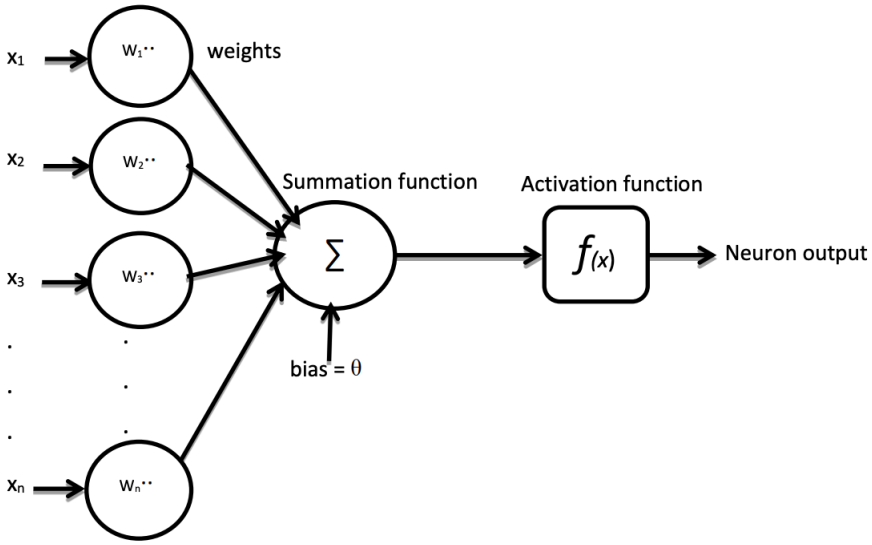


Figure 2.8: Simple artificial neuron, taken from [116]

The activation function of an artificial neuron controls the output of said neuron. There are many different activation functions available, some common activation functions are demonstrated by figure 2.9. The output of a given activation function, and hence the output the neuron, is dependent on the summed value. These activation outputs are nonlinear functions, several nonlinear activation functions chained together as part of a network allow the resolution of complex tasks. Selection of an appropriate activation function for each neuron in a network is dependent on the application and shall be discussed in the subsequent sections.

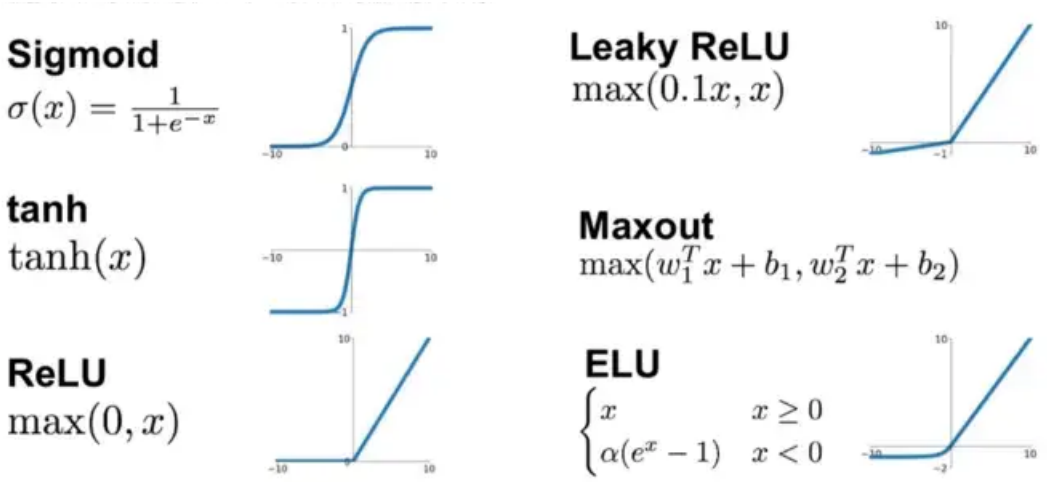


Figure 2.9: Common activation functions, taken from [117].

Construction of a neural network is achieved via arranging layers of nodes (neurons), an example neural network configuration is shown by figure 2.10. As demonstrated, a neural network consists of an input layer, an output layer and one or more hidden layers. Specifically, the configuration demonstrated shows a fully connected network whereby each neuron in a given layer is connected to all neurons from the previous layer. A fully connected neural network is made from nodes referred to as dense neurons given the density of connections. However, this is not the only type of neuron available as shall be discussed in later sections. The input layer does not manipulate the input in any way, instead it only distributes the sample to subsequent neurons. The hidden layer(s) is responsible for the majority of calculation and are difficult to interpret considering the high degree of interconnectivity between nodes [118] [119]. For simplicity, the hidden layers are often assumed to be a black box in which most of the computation is performed. Generally, the Rectified Linear Unit (ReLu) or variations of the ReLu function are a good choice for the activation function of these hidden neurons; however experimentation with other functions may also

yield good results. A network that includes 2 or more hidden layers is referred to as a deep neural network, increasing the number of hidden layers is associated with increasing a network's depth. Similarly, increasing the number of neurons per hidden layer increases a layer's width. The output layer attempts to collate the results from the hidden layer and predict an answer. The activation function of the output neuron(s) requires special attention as different applications (for example binary classification vs multi class classification vs regression) are relatively specific as to which function should be used. For example, for multi classification type problems the output neuron(s) must use the softmax function (variation of sigmoid). In addition, the number of neurons in the output layer should be equal to the number of classes present in the dataset. Conversely, the number of neurons in the input layer must be equal to the number of features of the dataset. The number of neurons in the hidden layer/the number of hidden layers is a hyper parameter that must be tuned in order to devise a model that performs optimally. Increasing the depth and width of a network increases the complexity hence can contribute to overfitting, the opposite is the case for under fitting.

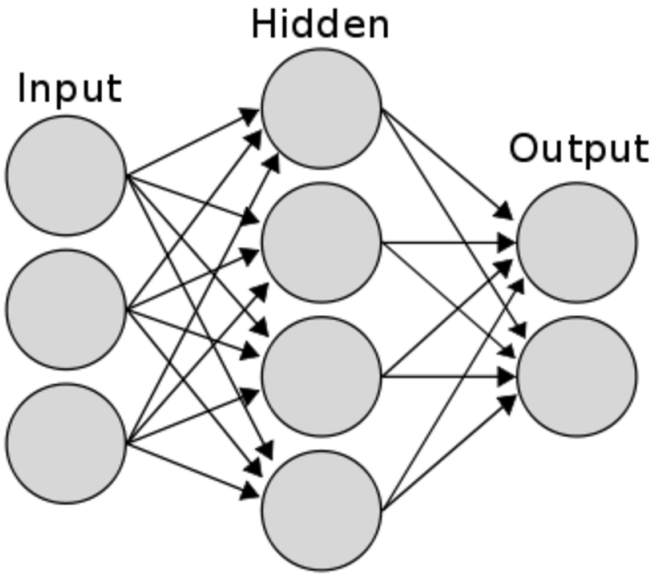


Figure 2.10: A simple neural network, consisting of an input, hidden and output layer. Taken from: [120].

Crucial to the training of neural networks, is the backpropogation algorithm which is an efficient method for calculating the gradient of the cost function, relative to the model parameters (weights and biases). For each training batch passed to the network, backpropagation makes a prediction (forward pass) and the error between true value and predicted value is calculated. Backpropagation then moves backwards through the network identifying the error contributions from all network weights (backward pass). Specifically, backpropagation calculates the error gradients for each neuron and associated parameters per sample, and averages these gradients per batch. An optimiser (such as gradient descent, see section 2.2) is then applied using the calculated gradient, to alter the network parameters to minimise the cost function. Note, initially the network weights are randomly initialised as this is expected by the gradient descent algorithm. Common issues experienced during the backpropagation process include vanishing, unstable and exploding

gradients. Vanishing gradients is when the gradients calculated during the gradient descent process, especially for the network layers closer to the input layer, become too small resulting in negligible weight updates for these neurons. This will in turn affect subsequent neurons in the network. In other circumstances, particularly when working with Recurrent Neural Networks (see section 2.4.2) gradients become so large they 'explode' and the algorithm diverges. The unstable gradients problem occurs when different layers of a neural network may learn at different rates. The solutions to these issues shall be highlighted in subsequent results sections.

Thus far, the only approach discussed for cost function optimisation has been gradient descent. However, neural networks have other techniques available for use (often extensions of gradient descent) that can significantly reduce network training time and improve network performance. For this work, when training neural networks, the adaptive moment estimation (ADAM) optimizer is favoured. The ADAM optimiser itself is a combination of two other optimisation techniques: momentum optimisation and RMSProp. Full details of which can be found in [99]. However a key concept of the ADAM optimizer is momentum. Momentum optimisation speeds up the learning rate as the minimum of the cost function is approached (gaining momentum) [99]. This is opposed to regular gradient descent which keeps the learning rate constant; therefore taking longer to converge to an optimal solution. On the contrary, learning rate scheduling is a technique that slows down the learning rate just as the descent algorithm approaches the optimal solution. This is done to prevent divergence from a solution when it is close to being achieved. There are multiple learning rate scheduling techniques available, however for this thesis performance scheduling is preferred. Performance scheduling reduces the learning rate if the validation data loss halts/increases over a defined number of training epochs. Overall, a combination of momentum and learning rate scheduling can significantly improve network performance

and reduce network training time.

Neural networks are prone to overfitting considering the often large number of model parameters associated with network configurations. To mitigate this, various regularisation techniques are available to prevent overfitting from occurring. For this work, 3 main ANN regularisation techniques are employed: dropout; weight constraint and early-stopping. Dropout involves randomly deactivating neurons in a network configuration resulting in reduced sensitivity to slight input changes; this simple strategy improves network generalisation. Note, dropout is applied during training, not whilst validating/testing. Weight constraint limits the size of network weights, this is important as large weights are associated with model overfitting. For this work, weight constraint is implemented via specifying a max norm of the weight vector per layer hence limiting the maximum weight values. Early stopping is a technique that identities when a network is overfitting and halts network training accordingly. This process works by monitoring the loss (cost) function of the validation data. If this loss function increases over a specified number of training epochs, the network is deemed to be overfitting and the early stop function is activated. This halts training preventing further overfit. The early stop control has a parameter 'restore best weights' which if set to true will revert the network back to its optimum conditions pre overfitting. Another import technique used in this thesis, for the training of neural networks, is batch normalisation. This is technique alleviates the vanishing/exploding gradients problem by adding an operation prior/after the activation functions of a network. Batch normalisation normalises the output/activation of each neuron per batch which smooths the optimisation process [121]. This reduces training time, decreases the significance of weight initialisation and somewhat limits overfitting [99].

Alternate neural network configurations, opposed to dense, have been briefly mentioned thus far. The remainder of this section discusses in more detail the other the configurations

used in this thesis: autoencoders; Convolutional Neural networks (CNNs) and Reccurent Neural Networks (RNNs).

2.4.1 Convolutional Neural Networks

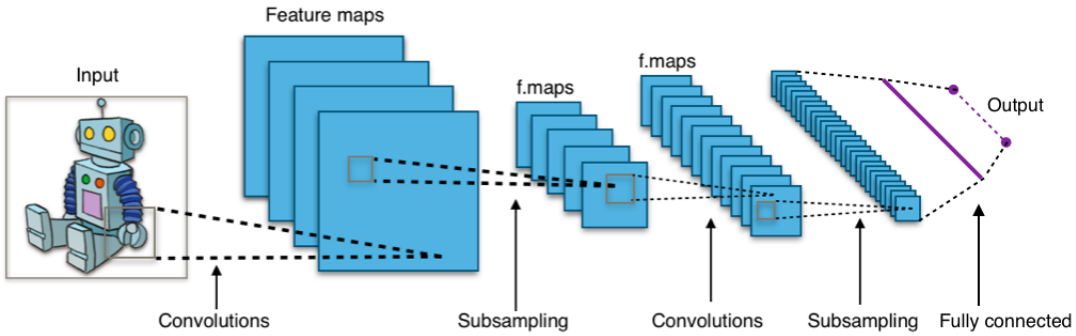


Figure 2.11: Example CNN structure taken from [122].

The design and implementation of Convolution Neural Networks (CNNs) is a direct result of the study of the brain's visual cortex. A CNN is built from convolutional layers that are ideally suited for tasks such as image recognition and sequence classification. Convolutional layers are similar to regular dense layers with regards to arrangement within a neural network (arranged in a sequential manner), however differ by the connections made between neurons. As discussed previously, fully connected/dense layers ensure a neuron in a given layer is connected to all neurons from the previous layer. This is not the case for convolutional layers where a neuron in a given layer is only connected to a subset of neurons, a receptive field, from the previous layer. This concept is demonstrated by figure 2.11.

An important CNN parameter is the stride which controls the spacing between the receptive fields of a convolutional layer. Increasing the stride will downsample an input as the increased spacing will result in fewer neurons in the subsequent layer. This process will

reduce the complexity of a CNN hence limit overfitting however can result in information loss if the stride is set too high. To ensure that a layer has the same height and width as the previous layer, it may be necessary to use zero padding. This pads the perimeter of a layer with zeros and is important if it is necessary to preserve the input dimensions of an image/sequence.

For a CNN to learn, network filters are required. CNN filters are a set of weights that connected neurons use to interpret an input. For each filter, a feature map is produced which is determined by the associated filter weights. The learning process adjusts the values of these weights within the filters in order to minimise the cost function. Each layer may have multiple filters which are set as a network hyperparameter. Naturally, increasing the number of filters increases the network complexity.

To reduce computational complexity, CNNs employ pooling layers to subsample the data. As with regular convolutional layers, each neuron in the pooling layer is connected to a receptive field of neurons from the previous layer. Depending on the configuration, a pooling neuron is set either to the max or average value of the activations from neurons in the receptive field. Note, there are no weights associated with the pooling process hence no 'learning' takes place for these layers.

2.4.2 Recurrent Neural Networks

RNNs are a specific class of neural network composed of recurrent neurons which are suited to dealing with sequence data [99]. This is because recurrent neurons incorporate information feedback as demonstrated by figure 2.122. Shown here is the simplest form of recurrent neuron, which at a given time-step receives the current input (x_t) and the output from the previous time-step (y_{t-1} or h_t, this may also be referred to as the hidden state [99]). Note, at time t=0, y_{t-1} is assumed to be 0 [99]. It is common to visualise a singular

recurrent neuron by 'unrolling' it through time, whereby: the input; output and hidden state per time-step are detailed as demonstrated by figure 2.122. From this example, it is clear how y_3 is a function of x_3 and h_2 (neuron output from previous time-step - the hidden state); and that y_2 is a function of x_2 and h_1 etc. Therefore, y_t is effectively a function of all previous inputs therefore introduces a form of memory hence recurrent neurons are often referred to as memory cells. It is this memory that allows RNNs comprised of layers of recurrent neurons, as opposed to just the singular recurrent neuron demonstrated as by figure 2.122, to interpret and learn complex signal/sequence patterns [99]. However, the rudimentary recurrent neuron shown by figure 2.12 suffers from short term memory whereby information is preserved only over a limited number of time-steps hence longer patterns cannot be realised [99]. To resolve this issue increased cell complexity is required.

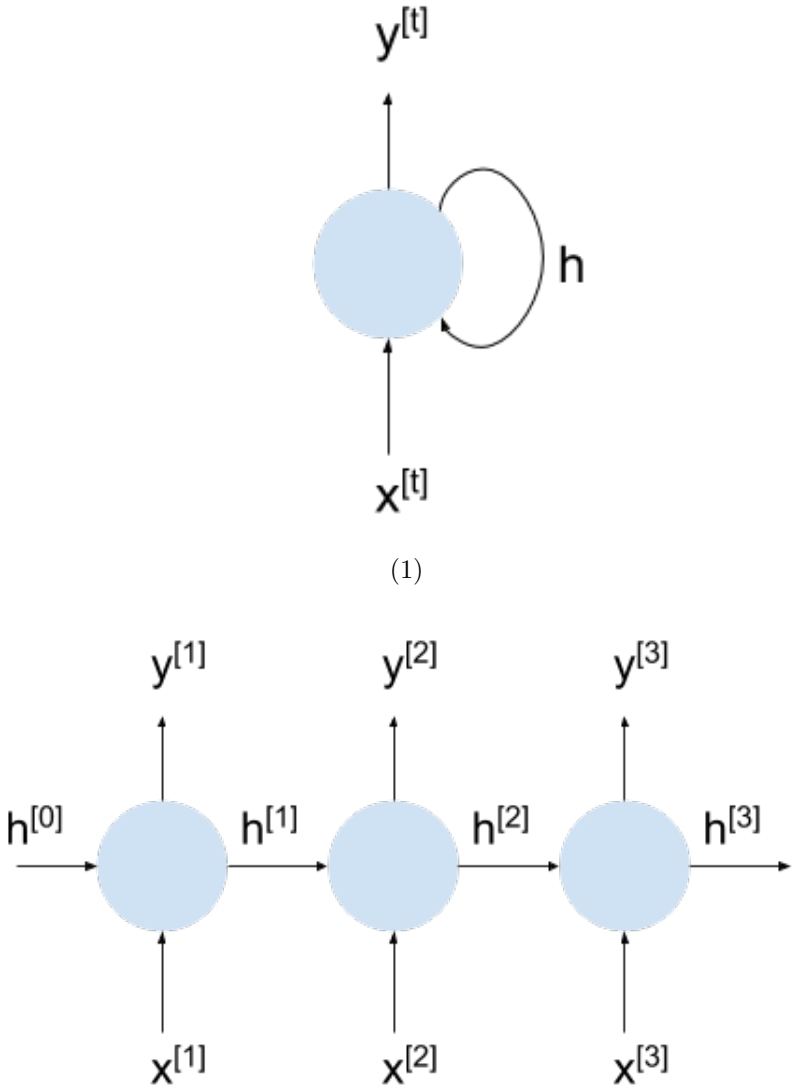


Figure 2.12: A Recurrent neuron. Figure 2.121 demonstrating how information is fedback and figure 2.122 showing how the cell is unrolled through time. Taken from: [123]

The Long Short Term Memory (LSTM) cell presents a solution to the short term memory issue of simple recurrent cells. Fundamentally, the LSTM not only utilises the

hidden (h_t) state but also incorporates a cell state (c_t). The hidden state may be considered the short term state and the cell state as the long term state. Crucially, a RNN of LSTM cells is able to decide what is important to keep in the long term state, discard from the long term state and read from the long term state; hence, increasing network performance when signal patterns exist over increased number of time-steps. The inner workings of the LSTM cell shall not be discussed in this work however it is important to understand the increased performance over longer sequences comes at a computational cost [99]. This can increase network training time which is already high compared to normal network neurons as a result of the feedback connections. A Gated Recurrent Unit (GRU) cell, devised in 2014 [124], is a simplified version of the LSTM cell that features fewer inner components however maintains similar performance compared to LSTM cells [99]. Hence, GRU cells are employed as the recurrent neuron of choice for the work detailed in this thesis. It should be noted that whilst LSTM/GRU cells attempt to minimise the short term memory problem, the issue is not eradicated therefore appropriate sample lengths must be selected.

To train a RNN backpropagation through time is performed, this process is identical to regular backpropagation however performed at all time steps for each neuron. Backpropagation through time significantly increases computational complexity hence RNNs take longer to train compared to standard feedforward networks for example.

2.4.3 CNN-RNN

A network constructed with a layer of 1D convolutional neurons, followed by a layer of recurrent neurons, can often achieve better performance with regards to sequence based tasks compared to a network comprised solely of recurrent neurons [99]. The covolutional layer outputs N (where N is number of convolutional filters) one dimensional signals and using a stride greater than one will reduce the size of the output sequence(s) from this

convolutional layer. This in turn reduces the size of the input sequence for the recurrent layer which helps to minimise short term memory loss.

2.4.4 Autoencoders

Autoencoders are a type of neural network that learn representations of data, typically in a lower dimensionality space. This makes this type of structure useful for tasks such as: dimensionality reduction or data visualisation. Autoencoders are comprised of two sub networks, an encoder and a decoder. These two networks are conical in shape as demonstrated by figure 2.13 with the encoder output having fewer neurons compared to the encoder input, the opposite is the case for the decoder. The number of neurons in the input/output layer should match the dimensionality of the training dataset.

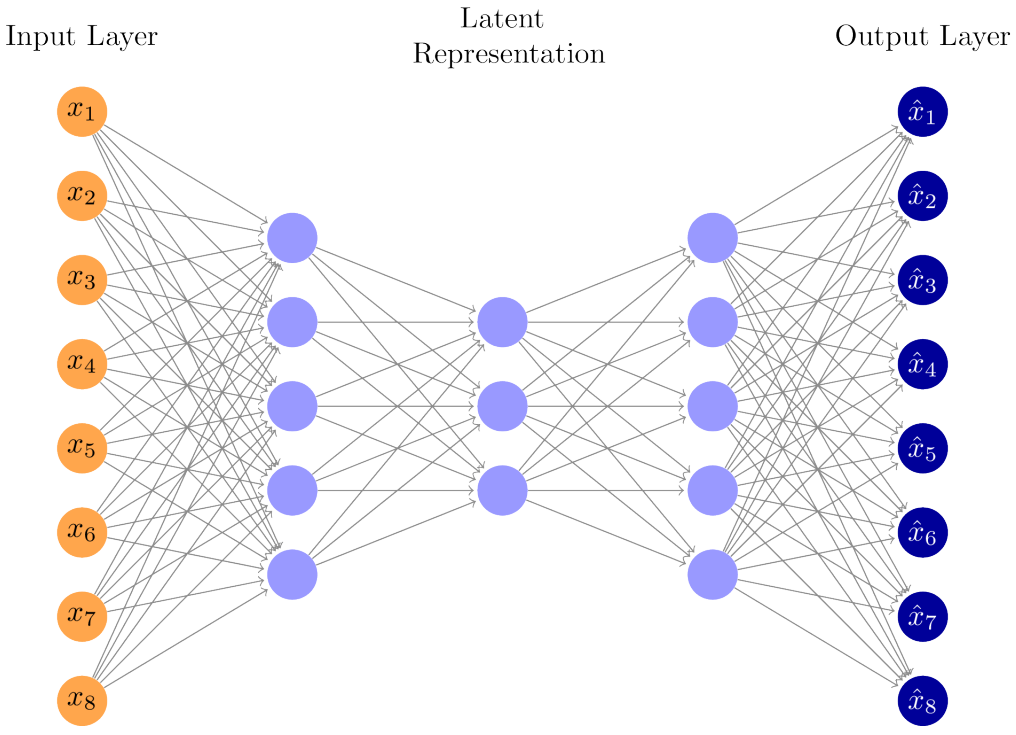


Figure 2.13: Example autoencoder structure. Taken from: [125].

Autoencoders during training learn how replicate the input (input of encoder) at the output (output of decoder). Once the autoencoder training is complete, the encoder can be separated and used to encoder higher dimensionality data to the lower dimensionality space. This new space will have dimensions matching the number of neurons present in the output layer of the encoder/input layer of the decoder. The benefits associated with using an autoencoder compared to PCA for dimensionality reduction is attributed the to non linear capabilities of neural networks. Compared to PCA which as discussed above is able to interpret linear relationships only. Note, in this thesis autoencoders are constructed using exclusively dense neurons, however can be implemented using convolutional/recurrent neurons.

2.5 State estimation Algorithms

2.5.1 Kalman Filter

The Kalman Filter is a recursive estimator of hidden variables based on inaccurate/uncertain measurements [126]. Fundamentally the Kalman filter is a data fusion technique that is based on the concept of the alpha-beta-gamma filter [127]. The principle of which is to fuse an estimate of the system state (the hidden variable) and a noisy measurement of the variable in question [128]. To achieve this three main concepts are necessary: a state estimate; a gain parameter and a state prediction [129]. In this thesis, as shall be discussed in later chapters, the hidden value is the BPM value from noisy ECG measurements.

Equations 2.18, 2.19, 2.20 detail the calculations required per step. For each step it is also necessary to predict certain values [126]. For this thesis, constant system dynamics are assumed. This is where the system state does not change. Equation 2.16 demonstrates this mathematically. Where $x_{n+1,n}$ is the estimate uncertainty for the next time step,

$x_{n,n}$, the current estimate uncertainty. To clarify, n here represents an iterative 'step' of the recursive process. For example, if the first step of the Kalman process were being calculated, $x_{n+1,n}$ resolves to $x_{2,1}$, i.e the next step relative to step 1.

$$\hat{x}_{n+1,n} = \hat{x}_{n,n} \tag{2.16}$$

Assuming constant system dynamics can result in high error, especially if the system state is changing (dynamic) [130]. However, it is possible to assume constant system dynamics (simplifying the Kalman process) even if this is not the case in reality. This is achieved during the uncertainty prediction process as described by eqauition 2.17. Where $p_{n+1,n}$ is the estimate uncertainty for the next time step, $p_{n,n}$, the current estimate uncertainty and q the process noise. The process noise (also normally distributed) is simply an uncertainty associated with the Kalman process.

$$p_{n+1,n} = p_{n,n} + q \tag{2.17}$$

The gain parameter used by the Kalman Filter is a special case which defines the algorithm. Equation 2.18 demonstrates the recursive process of the Kalman Filter (in 1 dimension as used in this thesis) where $\hat{x}_{n,n}$ is the state estimate (estimate of hidden variable) at time n, $\hat{x}_{n,n-1}$ is the previous state estimate, z_n is the current measurement and K_n is the current Kalman gain where $0 \leqslant K_n \leqslant 1$ [129].

$$\hat{x}_{n,n} = (1 - K_n)\hat{x}_{n,n-1} + K_n z_n \tag{2.18}$$

As demonstrated the Kalman gain controls how much the current state estimate ($\hat{x}_{n,n}$) relies on either on the physical measurement (z_n) or the previous state estimate ($\hat{x}_{n,n-1}$). Specifically increased Kalman gain adds more weight to the measurement and vice versa

[130]. Calculation of the Kalman gain is demonstrated by equation 2.19 where $p_{n,n-1}$ is the uncertainty of the previous estimate and r_n is the uncertainty of the current measurement measurement [129]. A crucial assumption of the Kalman Filter is that these uncertainties are distributed normally (Gaussian distribution) [130].

$$K_n = \frac{p_{n,n-1}}{p_{n,n-1} + r_n} \tag{2.19}$$

Having calculated the Kalman gain, the current measurement uncertainty can be calculated as demonstrated by equation 2.20. Where $p_{n,n}$ is the current measurement uncertainty and $p_{n,n-1}$ is the measurement uncertainty extrapolated from the previous timestep [129].

$$p_{n,n} = (1 - K_n)p_{n,n-1} \tag{2.20}$$

2.5.2 Conclusion

This chapter introduces the data and signal processing techniques that are used within this thesis. The implementation and required considerations where appropriate are also discussed. Signal/data processing is a crucial aspect of this work considering the emphasis placed on such methods instead of sensor selection/fabrication. As discussed in chapter 1, focusing on software solutions instead of hardware solutions is a key aim of this thesis which is pursed considering the associated reduction in cost and simplicity to deploy.

Chapter 3

Sensor System Design

3.1 Introduction

To construct a sensor system that is capable of monitoring physiological parameters in a range of environments, it is logical to design a multi sensor platform that can be easily modified for a given scenario, whilst maintaining the same basic configuration. This chapter discusses the design and construction of the fundamental sensor system constructed for this thesis. The modifications required relative to each deployment environment are explained and the methods used to generate basic physiological metrics (BPM, breathing rate etc) for each use case are outlined. These basic metrics are later interpreted in relavent chapters by more complex algorithms to generate information relative to the circumstance. As stated in chapter 1, a key theme for this work is the implementation of a wearable sensor platform mounted onto a chest strap. Such a device should be capable of operating in different extreme environments without requiring major electronic modification.

Naturally, some electronic/physical modification was necessary between the deployment environments discussed in this thesis (microgravity and underwater). In particular,

sensor system enclosure must be considered where appropriate, to prevent damage to the electronics. Logically, underwater use requires waterproofing which is not immediately necessary for microgravity applications. This difference, along with any further modifications required per environment, are discussed in the subsequent sections which detail the specifications required.

3.2 Design

3.2.1 Core sensor system

This section describes the design of the sensor system which remains consistent across all deployment environments. Fundamentally, all sensor systems constructed for use in this thesis are controlled using the Espressif ESP32 microcontroller [131]. This is a WiFi enabled device programmable via the Arduino IDE [132] that can communicate with sensors using various interfaces. Specifically, in this thesis, the ESP32 is configured to read/transport the sampled data via the Inter Integrated Circuit (I2C) interface [133], Serial Peripheral Interface (SPI) [134] and Analog to Digital Conversion (ADC) [135]. Selection of which protocol to use is dependent on the output of the sensor, hence the components and associated interfaces are detailed below. Note, the microcontroller is configured to sample data from the sensors at a rate of 200 Hz. The specifics of the ESP32 microcontroller are detailed by table 3.1, taken from datasheet [131].

Table 3.1: ESP32 specs

Parameter	Value
Supply voltage voltage	2.2V - 3.6V
Read Only Memory (ROM)	448 kByte
Random Access Memory (RAM)	520 kByte
Physical General Purpose Input Output (GPIO) pins	34
ADC resolution	12 bits
WIFI	802.11 b/g/n/e/l
Bluetooth	v4.2 BR/EDR and BLE
I2C	2 channels
SPI	4 channels

The remainder of this section shall provide an overview of each sensor used in this thesis, in addition some key specifications shall be indicated.

- ECG sensor - AD8232 [136]) single lead heart monitor. This device performs the necessary signal conditioning (see chapter 1) to produce readable ECG traces in standard conditions and outputs analog data which is read by the ADC on the microcontroller. The ECG electrodes used here are conductive rubber patches, fabricated into an elasticated chest strap. The strap used, and therefore electrodes, are sold commercially by Garmin [83] and intended for heart rate analysis during running. However, the connections to the electrodes can be exposed allowing electrical continuity to be established between electrodes and ESP32 ADC input. The specifics of the AD8232 ECG sensor are detailed by table 3.2, taken from datasheet [136].

Table 3.2: AD8232 specs

Parameter	Value
Common Mode Rejection Ratio (CMRR)	80 dB
Signal gain (G)	100
Supply voltage (V_s)	2V - 3.5V
Input voltage	0.2v - V_s

- Accelrometer - MMA8451 [137] 14/8 bit low power accelerometer. This device interfaces with the microcontroller via the I2C protocol and outputs acceleration values from x, y and z axes which in this thesis is useful for understanding body accelerations/movements. A specific application is the use of accelerometer data to estimate breathing rate when the device is positioned on/close to the chest. This is done by recording the movements associated with the expansion/contraction of the chest from which breathing rate can be calculated via FFT analysis. Naturally, this requires the subject to be stationary to avoid significant motion artefacts. For this work, only a single accelerometer is included on the sensor system. Initially an IMU, specifically BNO055 [27], was used however this device drew significant current causing the sensor system to fail. This is logical considering the multiple sensors featured and data fusion algorithms employed by an IMU device, hence a single accelerometer was favoured. Despite this, good results are demonstrated by subsequent chapters within this thesis as demonstrated by chapter 6. The specifics of the MMA8451 accelormeter sensor are detailed by table 3.3, taken from datasheet [137].

Table 3.3: MMA8451 specs

Parameter	Value
Select-able measurement range	(±) 2g/4g/8g
Supply voltage (V_s)	1.95V - 3.6V

- Temperature sensor - 10K thermistor [138] from which data is read via the ADC. The response from this sensor is lineralised using the natural logarithm in software. Usually further thermistor calibration is required to yield accurate temperature data, however for the purposes of this thesis only magnitude temperature changes are of interest The tolerance for the thermistor used here is ± 0.9% [138].

- Strech sensor - Conductive rubber stretch cord [139] the resistance of which changes depending on how much the cord is stretched. This cord used in series with a resistor of fixed value, forms a potential divider, the output of which can be read by the ESP32's ADC. This sensor, when orientated around the chest (attached to chest strap), can also be used to estimate breathing rate by recording the resistance changes associated with expansion/contraction of the chest. As with acclerometer data, this can be used to determine the breathing rate via analysis in the frequency domain however is more resilient to motion artefacts.

  Data is stored every ten seconds (2000 datapoints sampled at 200 Hz), by transferring all recorded data via WiFi to a local PC on which information was stored. This approach was selected as wireless communications grant freedom of movement and almost instantaneous information sharing.

  All electronics are housed within an appropriate enclosure mounted onto a chest strap (see figures 3.2/3.4). This strap is elasticated and features integrated ECG

electrodes to allow electrical analysis of the heart. The electrodes must be positioned appropriately (electrodes on front of chest, in line with the bottom of the sternum) so that ECG data can be read. As with the electronics, the strap used is a cheap, off the shelf product that is commonly used for cardiac analysis during sport/activity. The total cost of the electronics used is detailed by table 3.4.

Table 3.4: Cost of electronics

Component	Cost (£)
ESP32	19.50
AD8232	19.80
MMA8451	8.70
Stretch sensor	9.92
Thermistor	3.50
Total	61.42

The following sections detail the specifics of each sensor system vairant and any changes that are required to ensure functionality.

3.2.2 Sleep sensor system

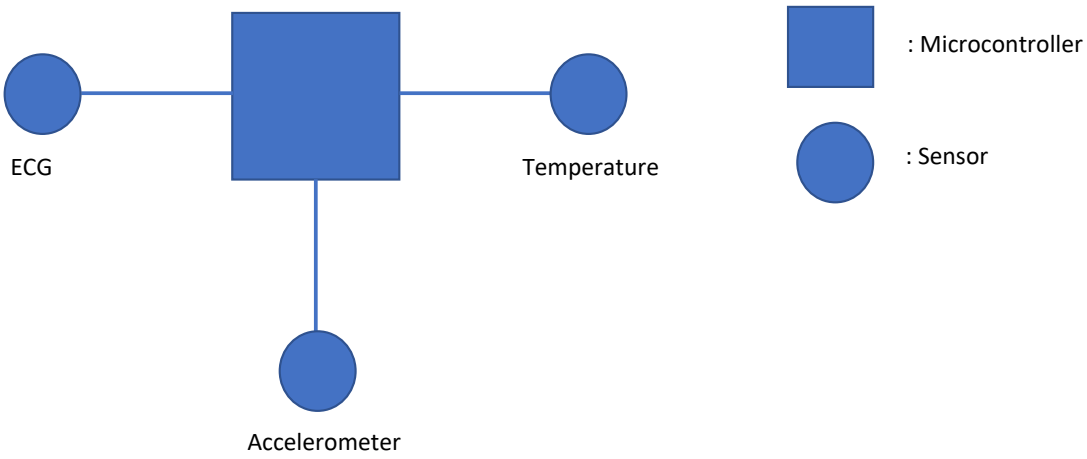


Figure 3.1: Block diagram of the constructed sleep sensor system variant.

Figure 3.1 demonstrates the block diagram detailing the electronic configuration of the sensor system variant used for microgravity applications; featuring the ECG, temperature and accelerometer sensors as per section 3.2.1. From these sensors BPM, breathing rate, skin temperature and acceleration (of the body) are calculated. All electronics were mounted onto a PCB and housed within a plastic enclosure.

Heart rate in BPM was calculated using MATLABs peak detection algorithm (as per section 2.1.1) where threshold and peak spacing values are set so to identify the R waves of the ECG waveform. Configuration of the peak detect algorithm varies between applications, generally minimum peak height is set at: 0.8 and minimum peak spacing: 0.5. This configuration works well in stationary conditions, however if heart rate significantly increases, the peak detection configuration must be adjusted. To verify the accuracy of the peak detection process, with regards to locating R peaks, visual analysis was performed comparing peaks identified by software against peaks identified by eye. Once the number of R waves is obtained for a given time frame, this value can be multiplied by an appropriate

factor (6 for a 10 second sample) to provide an estimate of BPM. This method is commonly used by medical practitioners as a quick approach to generate heart rate [9]. Generally, using this approach for estimation of any relevant physiological factor will result in quantisation issues, this is a significant consideration. However, for the results demonstrated in this thesis estimates are required on a relatively frequent basis hence such issues are noted but deemed unavoidable. This error could be minimised by increasing the sample length [140], which would limit this issue. However increasing the time window may be problematic in situations where regular physiological updates are required such as in hospitals or during sport. However, for this thesis, focus is generally directed towards the realisation of changes in physiological parameter magnitude, as discussed in subsequent chapters. Hence, it is more crucial that significant increases/decreases in these physiological parameters can be interpreted by any developed algoirthms. In addition, there are some algorithmic limitations associated, specifically with regards to ECG sample length with regards to BPM calculation. For certain cases in this thesis, the quantisation issues discussed above can be limited by averaging shorter (10 seconds for example) samples to minimise the resultant error. Particularly for the sleep focused chapters of this work which require less frequent metric generation. This averaging would not benefit metrics generated using FFT (cases discussed below), considering this process attempts to identify the dominant frequency rather than counting some quantity.

Breathing rate, for this system variant, was estimated from the accelerometer data. This was required because certain sleeping orientations limited the stretching of the cord sensor discussed in section 3.2.1, which resulted in incorrect breathing rate predictions. By isolating and analysing the z-axis of the accelerometer (axis facing outwards from the chest), over a 10 second period, a rhythmic signal is observed corresponding to the expansion and contraction of the chest. This signal can be analysed in the frequency domain using

FFT (2000 points, 0.1 Hz frequency resolution, rectangular window), where the breathing rate was estimated as the peak of the spectrum. This method performs relatively well in the absence of motion during sleep however is of course highly susceptible to motion artefacts. As before, validation of this approach was performed via visual comparison of the breathing rate estimations to the signal data recorded from the sensor system. To minimise motion artefact issues, upon detection of an obvious error (breathing rate estimates above 24 breaths per minute or below 7 breaths per minute), a mean breathing rate value is substituted for the erroneous estimate. Regarding the accelerometer data, the magnitude of the three accelerometer channels is calculated to provide an indication of the subject movement. The skin temperature data sampled from the thermistor is processed by calculating the mean generating an average value for each 10 second sample.

As discussed in section 3.2.1, every 10 seconds all sampled data is recorded. Samples are transmitted via WiFi (using a Transmission Control Protocol (TCP) socket) to a PC that is running a GUI (written in C++) that stores all of inbound data within text files on the computer. These text files are then analysed by the appropriate algorithms to generate sleep stage predictions.

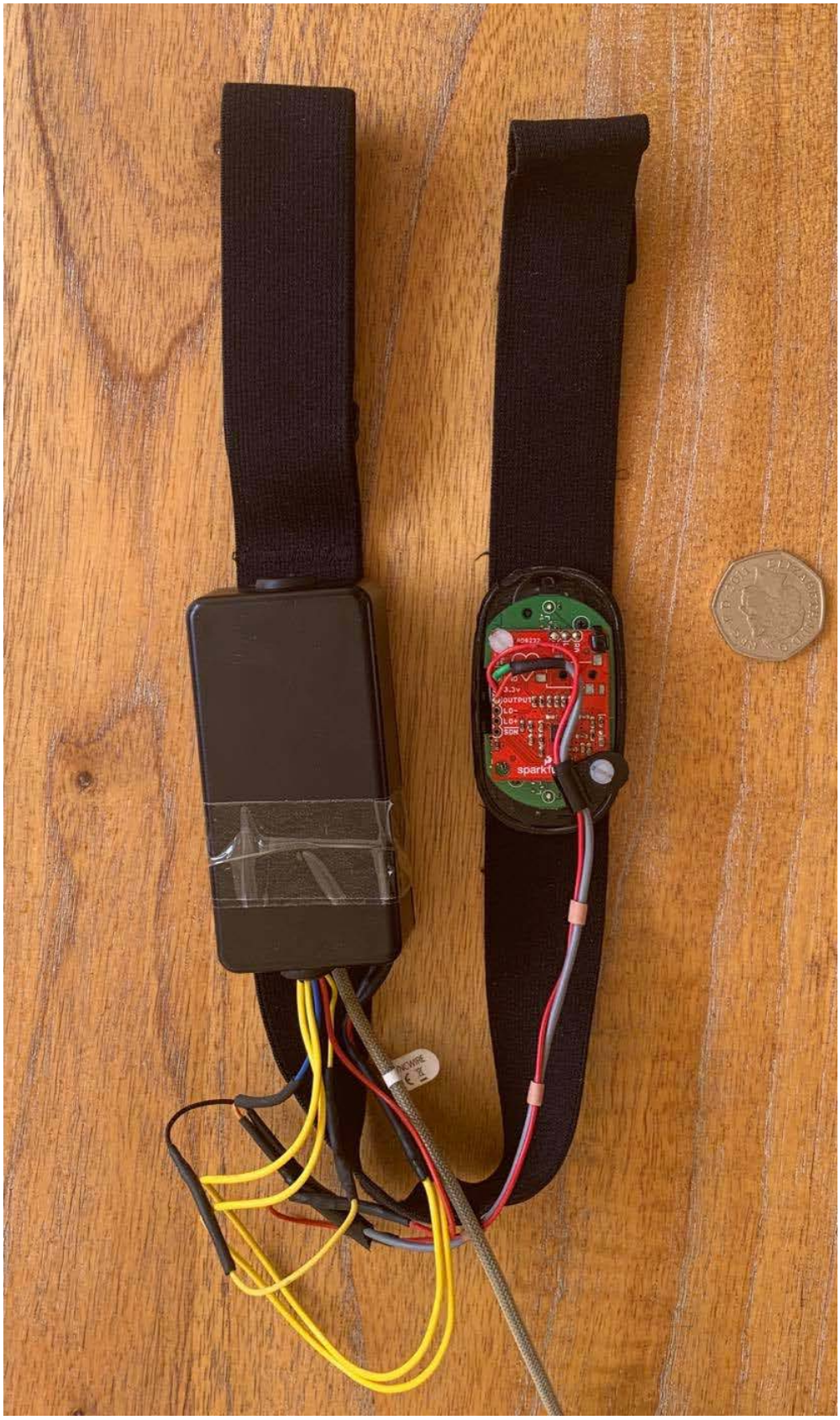


Figure 3.2: The constructed sleep sensor system variant.

3.2.3 Under water sensor system

Figure 3.3 demonstrates the block diagram detailing the electronic configuration of the sensor system variant used for underwater applications; featuring the ECG, breathing rate, temperature and accelerometer sensors as per section 3.2.1. From these sensors BPM, breathing rate, skin temperature and acceleration are calculated. As shown, a key difference here (compared to figure 3.1) is that a designated breathing sensor is required. This is due to the activities performed in the underwater environment, notably swimming, which renders estimation of breathing rate via chest accelorometer unfeasible due to motion artefacts. Instead, to determine breathing rate, a stretch cord (see section 3.2.1) attached to the strap monitors chest expansion and contraction. As with section 3.2.2, the resultant signal is analysed via FFT and breathing rate is estimated via peak identification of the spectrum. In addition to this, as demonstrated by figure 3.4, an IP67 waterproof enclosure [141] was used to house the electronics. Similar to the sleep system variant, the ECG electrodes are fabricated into the chest strap. However in this case the electrodes are connected to a specific sensor cable (an RJ45 Ethernet lead [142]). To provide a waterproof interface between electrodes (outside IP67 enclosure) and ECG sensor (within IP67 enclosure), an IP68 RJ45 junction is used [143]. The micro-controller is powered via a LiPo battery that is also housed within the enclosure.

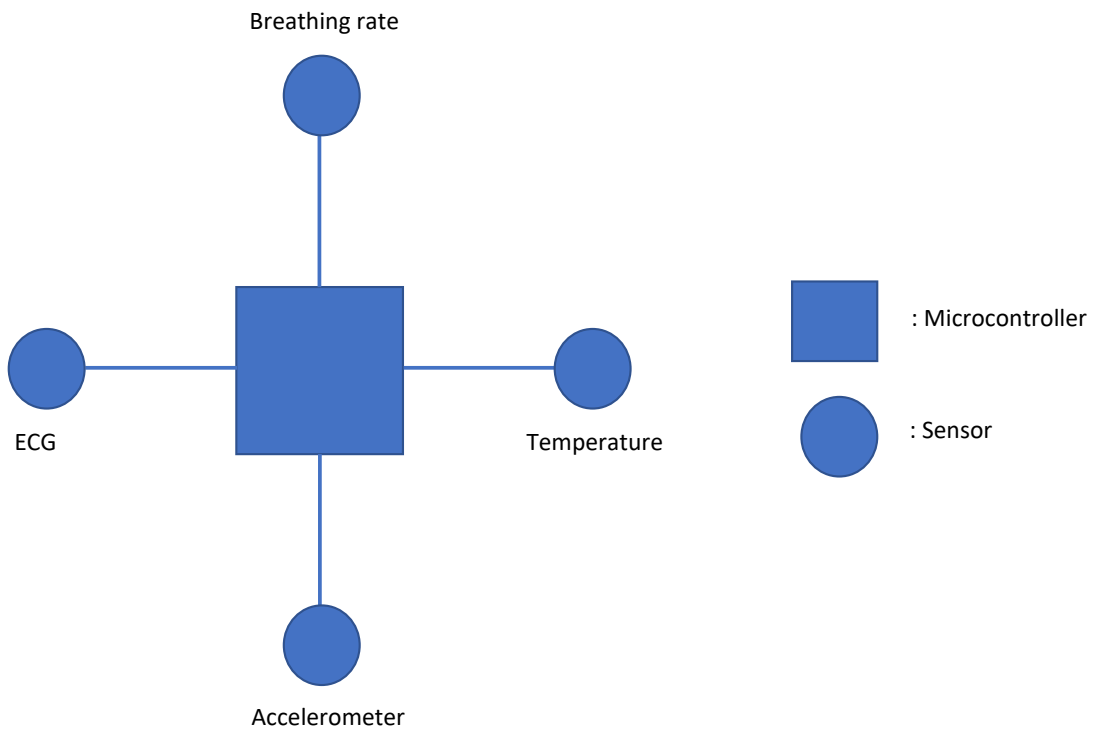


Figure 3.3: Block diagram of the constructed underwater sensor system variant.

As mentioned above, breathing rate is estimated via similar methods compared to the sleep system variant (spectrum analysis), albeit via use of a different senor. However, the BPM generation method requires major adaptation in order to predict realistic heart rate values. The reasoning for this is discussed in chapter 5, where the focus is the development of accurate underwater BPM generation algorithms. For analysis of underwater accelerometer data, the magnitude was not calculated as with section 3.2.2. Instead each individual channel was preserved and used for analysis. Although the temperature sensor was included on this sensor system variant, in this thesis temperature data is not included during underwater analysis. This is because for current applications this information is not required however will likely be beneficial for future analysis as discussed in chapter 6.

Figure 3.4: The constructed underwater sensor system variant.

During this work, the experiment faced significant challenges specifically when testing the system in a swimming pool due to the corrosive nature of the dissolved chlorine. The University of Liverpool swimming pool uses chlorine at concentration of 1.5ppm for sanitary purposes. Even at this relatively low concentration, damage was caused in particular to waterproof sealants, that after sustained testing would fail resulting in leaks within the enclosure/sensor cable. Fortunately, the LiPo battery used during this experimentation featured short circuit protection which automatically deactivated the battery upon water leakage. However, when water leaking occurred, all components within the enclosure would require cleaning to remove the resultant corrosion. The degrading sealant issue was resolved by selecting an enclosure (as per figure 3.4) with an appropriate seal (rubber). Limited options regarding enclosure were available considering the specific waterproof standard (IP67) and dimensions required (comfortably fit onto a chest strap). The option selected resulted in the need for careful PCB design to ensure that all components fit within the enclosure ensuring it could be sealed properly without resulting in damage. If this sensor platform were to be made into a commercial product, significant attention should be given to the waterproof/corrosive resistant housing. The orientation of the enclosure during swimming is demonstrated by figure 3.5. In addition to this issue, it is understood that Radio Frequency (RF) signals do not propagate well through water [144]. However, use of wireless data transmission is beneficial as discussed in chater 1, hence underwater WiFi transmission is investigated in chapter 4.

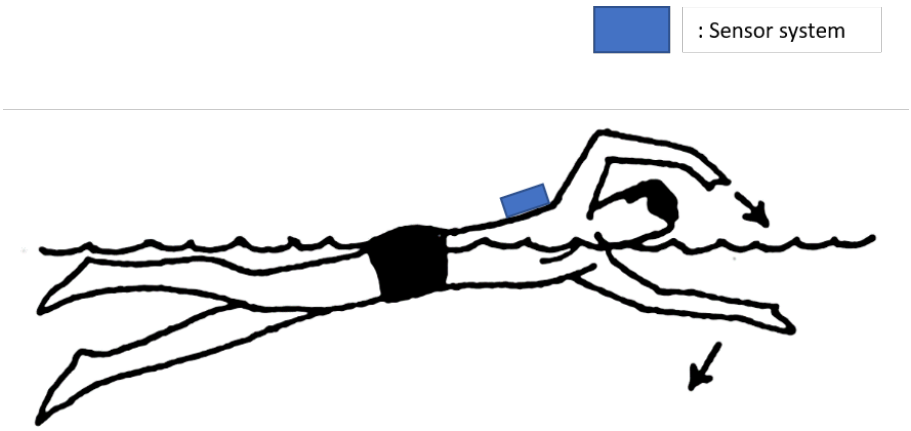


Figure 3.5: Orientation of the under water sensor system variant relative to a swimmer.

3.3 Conclusions

This chapter details the core sensor system used for the research detailed by this thesis. The two variants of this core system are discussed and the modifications required by each deployment environment are explained. Crucially the similarities are highlighted, specifically, the use of chest strap and sensors. The necessary modifications are then stated the most substantial of which being the requirement for a waterproof enclosure and the design considerations required. In addition, the generation of the basic physiological metrics used for analysis by specific algorithms is explained indicating when different techniques were required. The decision to construct a sensor system, instead of purchasing a pre-assembled unit (e.g. the EmotBit wearable biometric sensor tool), is due to the flexibility associated with this approach. This was important especially during early stages of design/testing when the optimal choice of sensor had not been determined. Dimensions of the PCB were also an important factor and required specific design, considering the spatial dimensions of the enclosures used in particular the waterproof variant. Lastly, the cost of preassembled

systems is usually higher, for example the EmotBit retails for $474.97 (USD). Significantly higher that the cost associated with the system designed for this thesis as per table 3.4.

Chapter 4

Transmission Through Water

4.1 Introduction

Extraction of data is an important step for the analysis of sampled information, particularly when considering real time analysis which requires use of wireless transfer methods. To implement this, there are several feasible techniques such as WiFi (IEEE 802.11 [145]), Bluetooth (IEE 802.15 [146]), Long Range (LoRa) [147] etc. As mentioned previously, for this work, WiFi is used given the increased data rate (compared to LoRa [147]) and increased range (compared to Bluetooth [148]). However this presents a series of challenges, particularly when used underwater as explored here. This chapter investigates transmission between a submerged transmitter, and a land based receiver as this reflects how the sensor system will be used in subsequent chapters. The effects of use in a swimming pool are also important to consider as the sensor system was tested in this environment. Studying this helped understand the feasibility of using WiFi to extract data in such conditions, no examples of which in literature were found. The key findings in this chapter are that underwater to above water WiFi transmission is likely to be limited in swimming pools

matching the conditions of the University of Liverpool pool. This appears mostly to be due the aqueous conductivity however in reality there are likely to be multiple factors.

Fundamentally, WiFi is a radio wave that has a frequency of 2.4/5 GHz (this work focuses of 2.4 GHz WiFi) and is commonly associated with internet communications. For such applications, WiFi communicates an internet connection between a router and a device requiring connectivity [149]. Also, thus far, all data processing has been performed offline however the use of WiFi for data transfer grants easy access to the live data from a remote Internet connection increasing telemedicine capabilities. However, the use of WiFi for data extraction purposes could be difficult in particular for the underwater setting through which Radio Frequency (RF) signals propagate poorly [150].This work demonstrates that transmission can be achieved up to 70 mm, however this is between submerged transmitter and receiver.In general, transmission of any electromagnetic wave through a medium is highly dependent on the conductivity of said medium. Specifically, increased conductivity increases the signal attenuation. Mathematically, electromagnetic propagation is described by equation 4.1 [151] where: α is the attenuation constant; β the phase constant and j is $\sqrt{-1}$.

$$\gamma = \alpha + j\beta \tag{4.1}$$

Specifically, α is described by equation 4.2 [151] where: ω is the angular frequency ($\omega = 2\pi f$, where f is frequency), μ the permeability of the medium, ϵ is permittivity of the medium and σ is conductivity of the medium.

$$\alpha = \omega \sqrt{\frac{\mu\epsilon}{2}\left(\sqrt{1+\left(\frac{\sigma}{\omega\epsilon}\right)^2}\right)}\frac{\mathrm{Np}}{\mathrm{m}} \tag{4.2}$$

β by equation 4.3 [151]:

$$\beta = \omega \sqrt{\frac{\mu\epsilon}{2}\left(\sqrt{1+\left(\frac{\sigma}{\omega\epsilon}\right)^2}+1\right)}\frac{\text{rad}}{\text{s}} \tag{4.3}$$

Comparing two underwater mediums: freshwater and seawater. Freshwater typically has a conductivity of: 100 µS/m whereas seawater typically has a conductivity of: 40000 µS/m. This is due to the Total Dissolved Solids (TDS) within seawater increasing the aqueous solution conductivity [152]. Because of the increased conductivity in case of seawater, for signal frequencies less than approximately 888 MHz, 4.4 is satisfied which according to 4.2 and 4.3 yields equation 4.5 [144].

$$\sigma >> \omega\epsilon \tag{4.4}$$

$$\alpha = \beta = \sqrt{\frac{\omega\mu\sigma}{2}} = \sqrt{\pi f \mu\sigma} \tag{4.5}$$

For such situations lower signal frequency and lower medium conductivity must be used in order to limit signal attenuation. Applying similar logic to fresh water scenarios: if 4.6 is satisfied and signal frequency is larger than a few MHz (as is the case for WiFi), equations 4.7 and 4.8 [144] define the attenuation and phase constants respectively.

$$\sigma << \omega\epsilon \tag{4.6}$$

$$\alpha \approx \frac{\sigma}{2}\sqrt{\frac{\mu}{\epsilon}} \tag{4.7}$$

$$\beta = \omega\sqrt{\epsilon\mu} \tag{4.8}$$

As demonstrated in this case, the attenuation constant is independent of signal frequency however remains dependent on conductivity.

For the RF applications discussed in this thesis, transmission requires an underwater transmitter to communicate with an above water router. Therefore such transmission involves a refraction boundary between air and water. Crucial to this concept is the consideration of a critical angle above which an incident electromagnetic wave is totally internally reflected. Such an angle is determined by first calculating the refractive index of the two mediums according to equation 4.9 [144] where: n is the refractive index; c the speed of light in a vacuum and v the speed of light in a given medium.

$$n = \frac{c}{v} \tag{4.9}$$

Having calculated the refractive index of both mediums in question the critical angle is calculated according to equation 4.10 [144] where ψ is the critical angle.

$$\psi = \arcsin \frac{n_2}{n_1} \tag{4.10}$$

Another important consideration with regards to electromagnetic transmission, underwater or not, is the path loss. Path loss is the reduction in power density for an electromagnetic wave as it propagates through space and is caused by multiple factors such as attenuation, absorption, reflection etc [153]. For this chapter, such calculation will consider: losses caused by underwater transmission; losses caused by above water transmission and any losses caused by the water-air boundary. Full details on such calculations are not detailed as part of this work, however a similar example involving underwater to above water electromagnetic transmission can be found in [144]. With these calculations performed, the received power can be calculated according to equation 4.11 [144] where: P_r

is the received power, P_t the transmitted power, G_r the receiver gain, G_t the transmitter gain and PL the net calculated path loss.

$$P_r = P_t + G_r + G_t - PL \tag{4.11}$$

As demonstrated by equation 4.11, the antenna gain of the transmitting and receiving devices are important considerations with regards to the received power. For this work, to provide an indication of the received power the Received Signal Strength Indication (RSSI) metric is used. RSSI provides an estimation of the wireless link quality which indicates the quality of the data packets received [154]. For the transmitting/receiving device (ESP32 microcontroller) used during these experiments, the value of RSSI ranges from -30 dBm to -90 dBm. With values closer to -30 dBm indicating a good signal strength. Table 4.1 [155] details this range in more detail. The antenna used here (and in all other chapters), is the standard PCB antenna fabricated onto the microcontroller by Espressif [131]. The ESP32 supports IEEE 802.11b/g/n, here 802.11n is used. Espressif do not state if MIMO is used by the specific ESP32 version (ESP32 WROOM [131]), and no evidence can be found online to confirm or deny this. This is also the case for the antennas directionality.

Table 4.1: RSSI levels explained.

RSSI (dBm)	Signal quality
-30	Max achievable signal strength
-67	Minimum required signal strength for applications such as streaming
-70	Minimum signal strength required for reliable packet delivery
-80	Minimum signal strength required for packet delivery
-90	Signal indistinguishable from noise. Functionality unlikely

This chapter considers how the submersion of an RF transmitter affects data transmission to an above water base station, as is the case when using the underwater sensor

system variant. Specifically, during this experimentation, the transmitter is submerged in water where depth, conductivity, the presence of specific chemicals are varied and the resultant RSSI values between transmitter and router are recorded. In addition, the affect of absorption due to the body is also observed. This experimentation was done to investigate the feasibility of under to above water transmission for later experimentation and considers how multiple factors might be relevant to later deployment.

4.2 Methods

4.2.1 Data acquisition process

To obtain the results discussed in this chapter, a sensor system configuration was used which is not described in chapter 3. This is because no physiological measurements were necessary for this experimentation, instead only the recording of RSSI values between the ESP32 (submerged transmitter) and WiFi router. Hence only an ESP32 microcontroller (powered by a LiPo battery), WiFi router and control GUI were required. To prevent water damage, the microcontroller was housed within a sealed Tupperware box. The microcontroller measures the RSSI value whilst connected to the WiFi router, indicating the signal strength between the two devices. The transmission gain was set as 19.5 dBm which is the maximum value for the ESP32 microcontrolle. This will likely increase the power consumption of the microcontroller, although no instances of the LiPo battery fully discharging were encountered.

4.2.2 Experimental setup

Figure 4.1 details the experimental setup used to generate the results discussed in this chapter. As shown, the microcontroller is positioned in a water tank. This tank is shielded

using foil so that RF signals are attenuated when propagating out of the sides/bottom of the tank. Shielding is used here to mimic the concrete structure of a swimming pool. The tank can be filled with water to understand the effect this has on the RSSI, the distance between router and tank was fixed at 1.5m.

As discussed previously, experimentation involves varying depth, conductivity of the aqueous solution, presence of chemicals/absorption material to observe how these factors affect the RSSI. Specifically, two depths are investigated: half and full tank depth (14.5 cm & 29 cm). Aqueous solution conductivity, at these distinct tank depths, is increased incrementally with the RSSI measured at each step. Also at each depth, independent to the conductivity experimentation, chlorine dissolved in the water matching the concentration used by the University of Liverpool swimming pool (1.5 ppm) is investigated. A minimum of twenty RSSI values were recorded per depth/salt concentration increment the mean of which was calculated.

To understand the effect of electromagnetic absorption caused by flesh, this factor was investigated by attaching pork belly the transmitter. This was done in attempt to understand whether the body significantly attenuated the signal. Here, pork meat used as an analog for human muscle/fat. All experimentation has been performed witihn room G/053 in the University of Liverpool, EEE B block. This room is not shielded from external electromagnetic radiation hence RF interference is likely to be a factor. However, in interest of gaining a rudimentary understanding of underwater RF transmission, this interference was deemed not to be a critical consideration.

Experiments involving conductivity were controlled by dissolving salt in tap water, increasing aqueous solution conductivity. The salt concentration is increased incrementally (3.5 g/L) per increment. As mentioned above, at each salt concentration the conductivity is measured with a Total Dissolved Solids (TDS) probe, and the value compared to a mean

RSSI value determined by the microcontroller.

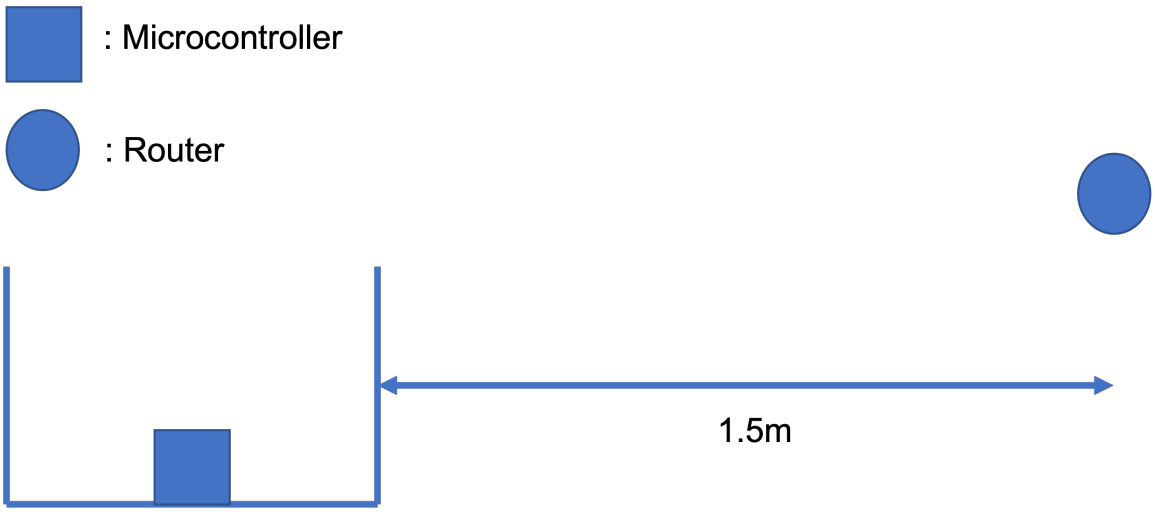


Figure 4.1: Diagram of the experimental setup used to generate transmission results.

4.3 Results & Discussion

Table 4.2 details the mean RSSI values recorded with the microcontroller submerged in tapwater at half and full tank depth. As shown, when the tank is filled to its maximum depth the RSSI value indicates a weaker signal strength between transmitter and router. This is expected considering the larger volume of water will result in increased electromagnetic path loss (more attenuation, absorption etc) as discussed in section 8.1. The mean RSSI value recorded (-81.25 dBm), is close to the minimum value required for packet delivery and therefore a weak connection as per table 4.1.

Table 4.2: Mean RSSI between transmitter submerged in tap water at half and full tank depth, and land based router.

Depth	Mean RSSI (dBm)	Standard Deviation
Half depth	-67.25	±2
Full depth	-82.56	±1

Table 4.3 details the mean RSSI values recorded with the microcontroller submerged in chlorinated tap water at half and full tank depth. As demonstrated, the presence of chlorine has little affect on the RSSI values compared to un-chlorinated tap water. Unexpedtedley, the results in chlorinated water indicate slightly improved RSSI values. However, this improvement was not further investigated given the minor change observed.

Table 4.3: Mean RSSI between transmitter submerged in chlorinated tap water (chlorine concentration: 1.5ppm) at half and full tank depth, and land based router.

Depth	Mean RSSI (dBm)	Standard Deviation
Half depth	-65	±3
Full depth	-81	±2

Table 4.4 details the mean RSSI values recorded with the microcontroller submerged in tap water at half and full tank depth. For this test, pork belly was attached to the transmitting device in attempt to understand the attenuating affects caused by human muscle/flesh. As demonstrated, this had little affect on the RSSI values compared to the results demonstrated by table 4.2. Assessment of standard deviations indicate that the different conditions investigated are not significant considering that the standard deviations are larger than the magnitude differences.

Table 4.4: Mean RSSI between transmitter (signal attenuated by pork belly) submerged in tap water at half and full tank depth, and land based router.

Depth	Mean RSSI (dBm)	Standard deviation
Half depth	-68	±3
Full depth	-78	±1

Having investigated this, water conductivity was the next factor considered. The conductivity of pool water is increased compared to tap water, this is due to multiple reasons one being the salt dissolved from human skin. As discussed in section 4.1, water conductivity is correlated to the electromagnetic attenuation hence was investigated as per section 4.2.2. Figure 4.2 details the effect of increasing the aqueous solution conductivity has on the mean RSSI value found between the submerged transmitter and router. The initial unsalinated reading is marked with a 'o' compared to subsequent 'x' markings indicating the addition of salt. Uncertainty bars are included on this graph considering the twenty measurements recorded per conductivity value. As shown, there is a general downwards trend when adding salt from the initial recording, thus indicating a decreased signal strength with increasing the conductivity of the water.

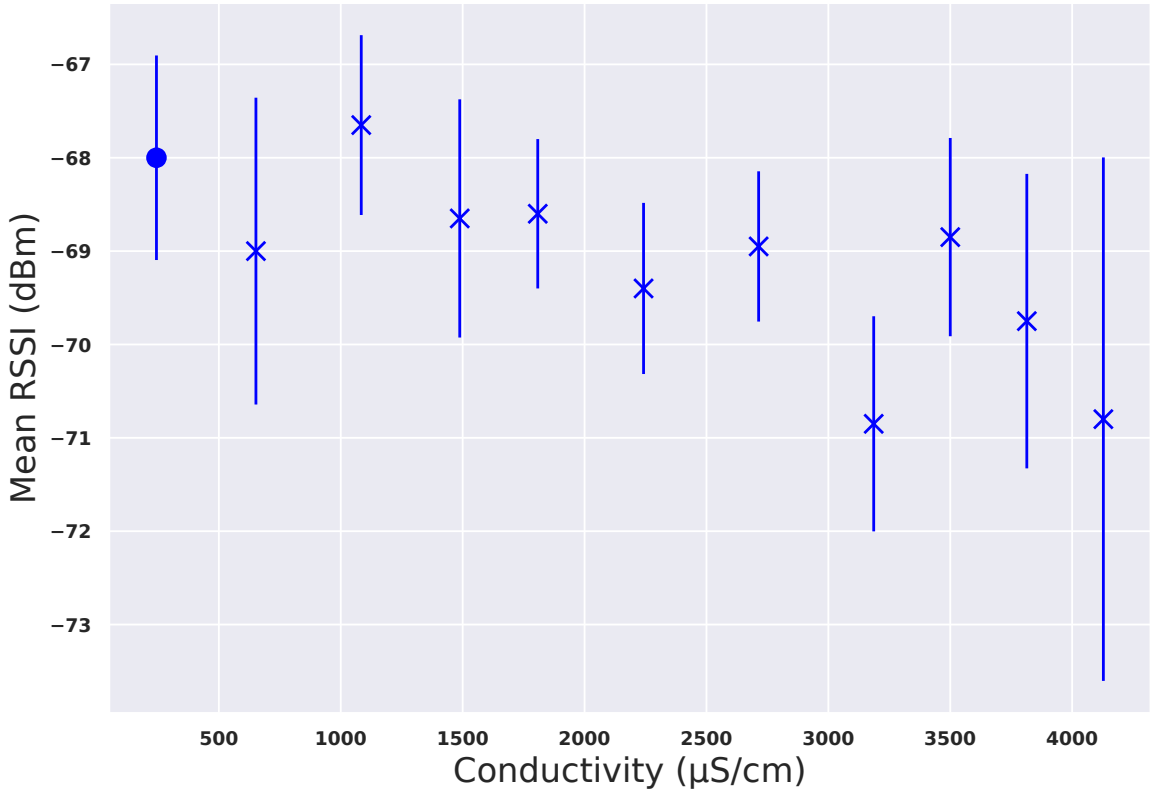


Figure 4.2: The affect of varying solution conductivity (by increasing salt concentration), on the mean RSSI value between submerged (half tank depth) transmitter and land based router.

Figure 4.3 details the results on the mean RSSI between submerged transmitter and land based router. However, in this case the transmitter is submerged to full tank depth. The results in this case do not indicate a linear downward trend, instead it initially increases before decreasing. The cause of this is not understood and it does not match the linear downward trend demonstrated by figure 4.2. As a hypothesis, it is possible that this trend is due to random environmental/multipath factors. However, this was not further investigated as it is beyond the scope of this investigation. What is demonstrated is that at maximum conductivity (approx 4000 µS/m), the mean RSSI was found to be -90dBm

which is unsuitable for packet delivery as indicated by table 4.1.

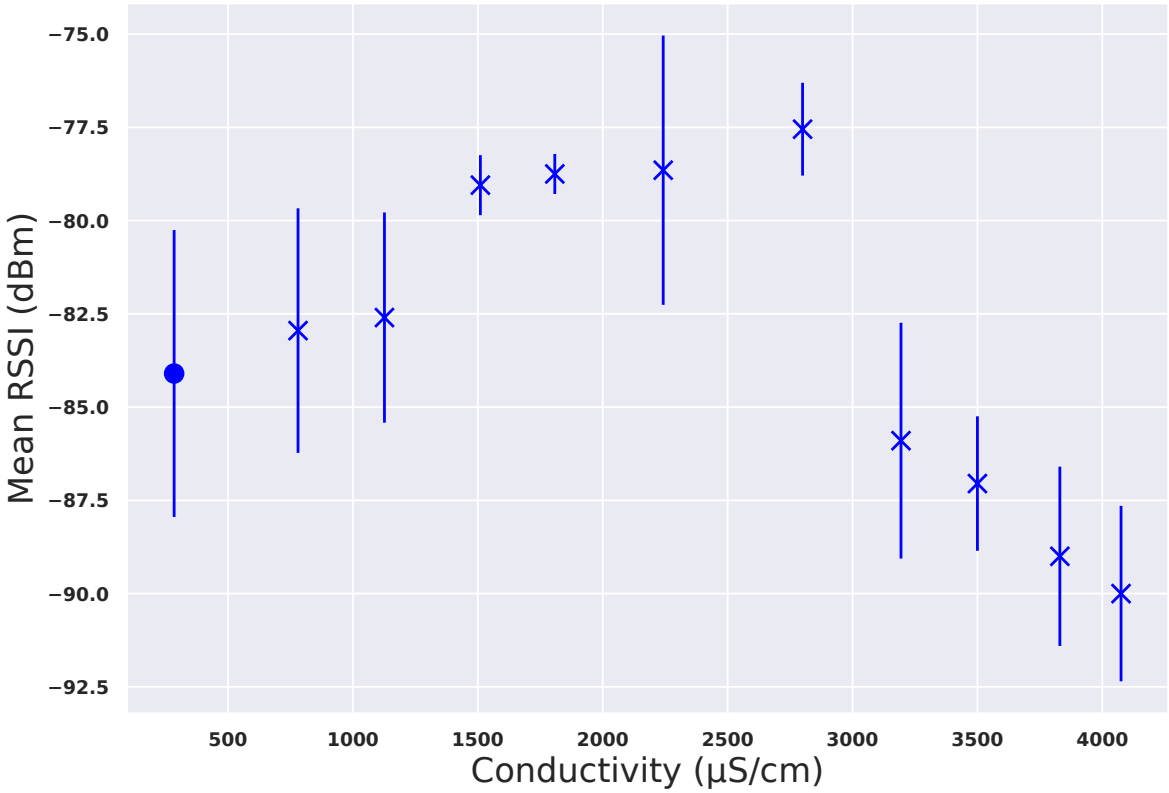


Figure 4.3: The affect of varying solution conductivity (by increasing salt concentration), on the mean RSSI value between submerged (full tank depth) transmitter and land based router.

Given that swimming pools have a conductivity range from 2500 to 3000 µS/cm, it is feasible that this will have a significant impact on data transmission. The above demonstrated RSSI values noted at the corresponding conductivity's for half and full depth experimentation, are approximately -71 and -86 respectively. Both of which correspond to a poor connection indicating there will likely be transmission issues at relatively shallow depths. Also, there are additional factors not investigated in the chapter such as the separation between transmitter and receiver. During this experimentation separation is fixed to 1.5m,

increasing this separation will likely affect the RSSI between submerged microcontroller and land based router. Understanding the significance of this affect is a relatively straightforward process by repeating the above investigations however at increased transmissions ranges. Finally, the affect of swimming pool structure is another factor not considered. Notably the metal used for the foundations of the pool may have an affect on radio transmission. However, unlike transmission separation, this factor would be significantly more difficult to understand given the experimental setup that would be required.

4.4 Conclusions

This chapter investigates the potential difficulties associated with the use of WiFi in underwater environments. It seems, as expected, that water conductivity has a significant affect on the RSSI and therefore the reliable transmission of data. In reality, a full understanding of under to above water transmission is more detailed as many factors will have an affect, some of which are beyond the scope of this thesis. However, gaining of rudimentary understanding proved somewhat useful as it was indicated that some limited transmission may be achievable. In future, use of other transmission protocols such as LoRa could be investigated considering the lower transmission frequency compared to WiFi. This may be beneficial considering the relationship between attenuation and transmission frequency as per section 8.1. Also, investigation of higher power transmitters in an attempt to reduce path loss could be another useful consideration.

Chapter 5

Underwater ECG Analysis

5.1 Introduction

Having initially investigated sleep analysis which focused on microgravity applications, it was determined that the sensors used by the constructed device were suitably sensitive to identify the small physiological changes associated with the different sleep stages. Hence, underwater investigations were performed to assess sensor response to larger signal variations. Chapter 1, section 1.3.6 discusses the difficulties associated with sensing in underwater environments and the challenges particularly with regards to physiological measurement. This chapter focuses on the analysis of the underwater ECG signals recorded by the sensor platform which has many applications such as: performance swimming; rehabilitation and/or divers in hazardous environments. Also discussed in chapter 1, section 1.3.6, is the difficulties associated with existing solutions to this problem primarily electrode design methods and signal processing techniques.

As discussed in chapter 1, section 1.2, BPM is a crucial metric observed from ECG data hence is the focus here. To achieve this, four software based time domain solutions

are investigated. Peak detection (chapter 2, section 2.1.1), Continuous Wavelet Transform (CWT - chapter 2, section 2.1.3), a 1 dimensional Kalman Filter (chapter 2, section 2.5.1) and three neural network configurations are demonstrated. Specifically, the use of any additional signal filtering post sensor output, is avoided in an attempt to minimise information loss which is prevalent for ECG filtering applications [156]. This is achieved via the software techniques highlighted above, the best performing of which being a neural network comprised of both convolutional (chapter 2, section 2.4.1) and recurrent (chapter 2, section 2.4.2) neurons. All methods are designed to count the number of beats over a fixed sample length (10 seconds). The purpose of this is to scale the count (multiply count by 6) to estimate BPM, which is a method used by medical practitioners to quickly generate BPM values [9]. However, this approach gives rise to BPM error which can be minimised by increasing the sample length [140], but this will result in recurrent short term memory losses, as per chapter 2, section 2.4.2. A solution would be to omit the use of recurrent neurons for which short term memory is an exclusive issue, however the best beat counting performance is achieved when recurrent neurons are included. An alternative solution would be to reduce sampling rate, but increase sample length so that the number of samples remains constant. Whilst this would increase sample length, hence reduce BPM calculation error, reducing the sample rate below certain values can cause R peak distortion [157]. As stated above, a CNN-RNN model achieves the best performance with regards to this task. Which is logical considering such a configuration utilises recurrent neurons whilst minimising short term losses as per chapter 2, section 5.2.6. All approaches require minimal signal preprocessing prior to application and in addition do not require specific physical design either to: electrodes; amplifiers or chest strap in order to estimate BPM from underwater ECG signals. Crucially, as stated in chapter 1, this work has been accepted for publication in the journal: Applied Intelligence [2].

5.2 Methods

5.2.1 Data acquisition process

The sensor system variant used to generate the results discussed in this chapter is detailed by chapter 3, section 3.2.3. All data presented has been obtained from a single user, in accordance with relevant ethical approval processes in place at the time of testing. Figure 5.1 demonstrates an example land sampled ECG trace that has been resampled at 50 Hz from 200 Hz.

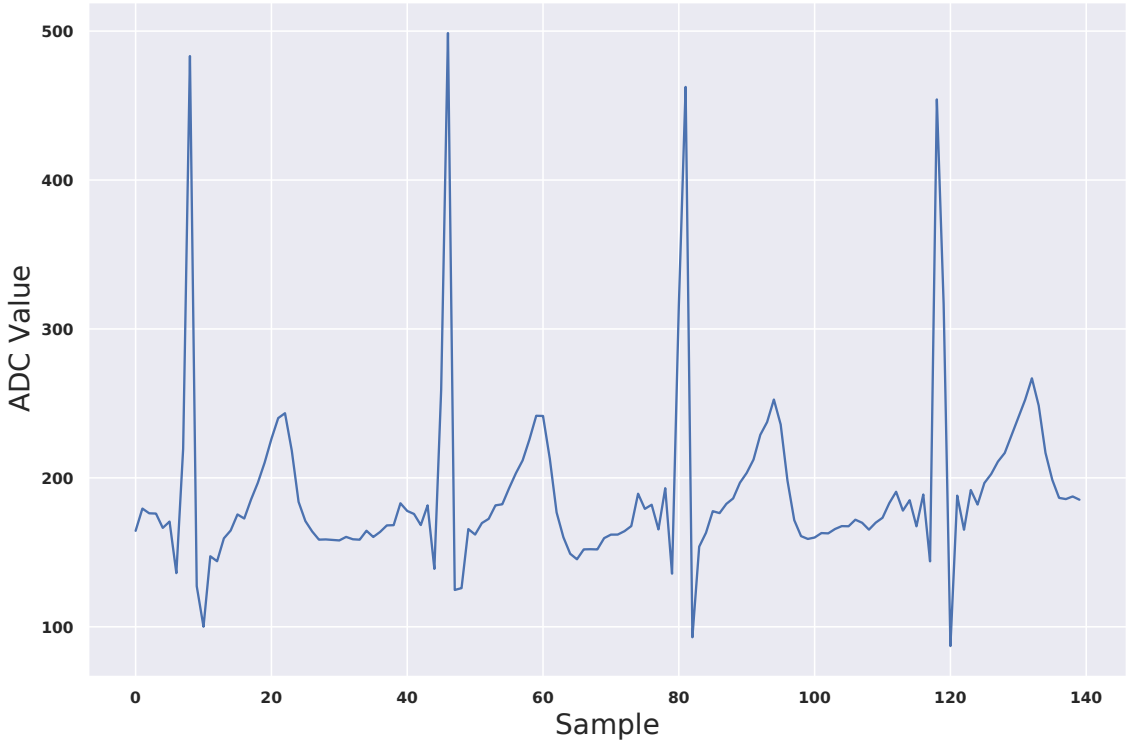


Figure 5.1: Example ECG trace resampled at 50 Hz.

This device has been used to sample 525, 10 second ECG samples have been collected either from a swimming pool (chlorine concentration 1.5ppm) or bath (UK tap water).

Different activities such as treading water or front crawl swimming were performed during data collection to generate a range of BPMs. These samples were used to assess the performance of the data processing techniques developed in this chapter and are normalised to a range (-1,1). Initially underwater ECG sequences were sampled from a bath as this approach grants a controlled, accessible environment from which sensor/transmission functionality could be verified. Experimentation then progressed to a swimming pool in order to allow dynamic movements such as swimming. In both cases, for the discussed activities, data extraction via WiFi was successful given the limited depth of submersion during front crawl/treading water. This is in agreement with the analysis performed in chapter 4.

5.2.2 Peak detection configuration

It was expected that the peak detection configuration used to locate R peaks from clean ECG signals, as discussed in previous chapters, would not be suitable for application to noisy underwater ECG samples. Hence the algorithm settings were altered in attempt to achieve better performance. Table 5.1 details the configuration used to achieve best results. These peak detection settings are also used to generate the noisy measurements used by the Kalman Filter approach discussed later in this chapter.

Table 5.1: Peak detection configuration.

Parameter	Value
Peak height	0.4
Peak spacing	20

5.2.3 CWT configuration

The CWT configuration used in this chapter matches that which is described by MATLAB for the purpose of locating R peaks from noisy ECG samples [95]. The CWT specifications are described by table 5.2. As stated by MATLAB, the sym4 wavelet is selected as it resembles the shape of the QRS complex and scales 4 & 5 are used to cover the passband frequency maximising QRS wave energy.

Table 5.2: CWT configuration.

Parameter	Value
Wavelet	sym4
Scales	4 & 5

5.2.4 Kalman Filter configuration

The Kalman Filter is a recursive process that has multiple parameters which require initial conditions to be set prior to application of the estimator. The values that give the best performance are highlighted by table 5.3. The measurement uncertainty has been set as the square of the MAE (the variance assuming errors are normally distributed as per chapter 2, section 2.5.1) between measured peak detect beat counts and true beat count values for the underwater ECG samples. The initial estimate is set to 15 which equates to 90 BPM (10s sample). This initial estimate is set considering the activities being performed during underwater ECG sampling.

Table 5.3: Kalman Filter fixed initial conditions.

Parameter	Value
Measurement uncertainty	23.671
Initial estimate	15
Process noise uncertainty	0.25
Estimate uncertainty	2.25

5.2.5 Pan Tompkins Algorithm configuration

The Pan Tompkins Algorithm is configured with sampling frequency 50 Hz. This is chosen to match the sampling frequency of the underwater ECG data (post resampling).

5.2.6 Neural networks configuration/hyperparameters

Three neural network architectures are considered in this chapter. Specifically: CNN; RNN and CNN-RNN models are investigated. All are constructed using the python library Tensorflow Keras [158], the structures of which are detailed below. The configurations shown (number/width of layers, dropout, Pooling etc) achieve the best performance during validation.

Certain parameters are kept constant across all models. Firstly, the ADAM optimisation algorithm is used with a MSE cost function for regression purposes. Secondly, during training, for all models the batch size was set at 32 and the learning rate of the ADAM optimisation algorithm is configured to be 5e-4. However, this is lowered during training by a ReduceLROnPlateau callback function. This callback function has been configured with: reduction factor set to 0.5 and patience (number of epochs pre calling function) set to 5. Thirdly, to prevent exploding gradients, the normal of the gradient vector for the

optimisation algorithm is clipped to 1 and lastly, early stopping is used for all models as a method of regularisation with a patience of 10 epochs. Also consistent across all models is a single dense output neuron. The output of this neuron is rounded to give an integer value (R-peak count). All neural network methods discussed in this section are detailed in chapter 2, sections 2.2/2.4.

CNN

The structure of the CNN model is detailed by figure 5.2. As shown two convoluational layers are used, the specific hyperparameter configurations of which are identical and detailed by table 5.4. A dropout layer (dropout: 50%) is used after these convoluational layers followed by max pooling layer (pool size: 2). The output from which is flattened prior to processing via a dense layer which is configured according to table 5.5. A dropout layer (droupout: 10%) is used after this dense layer.

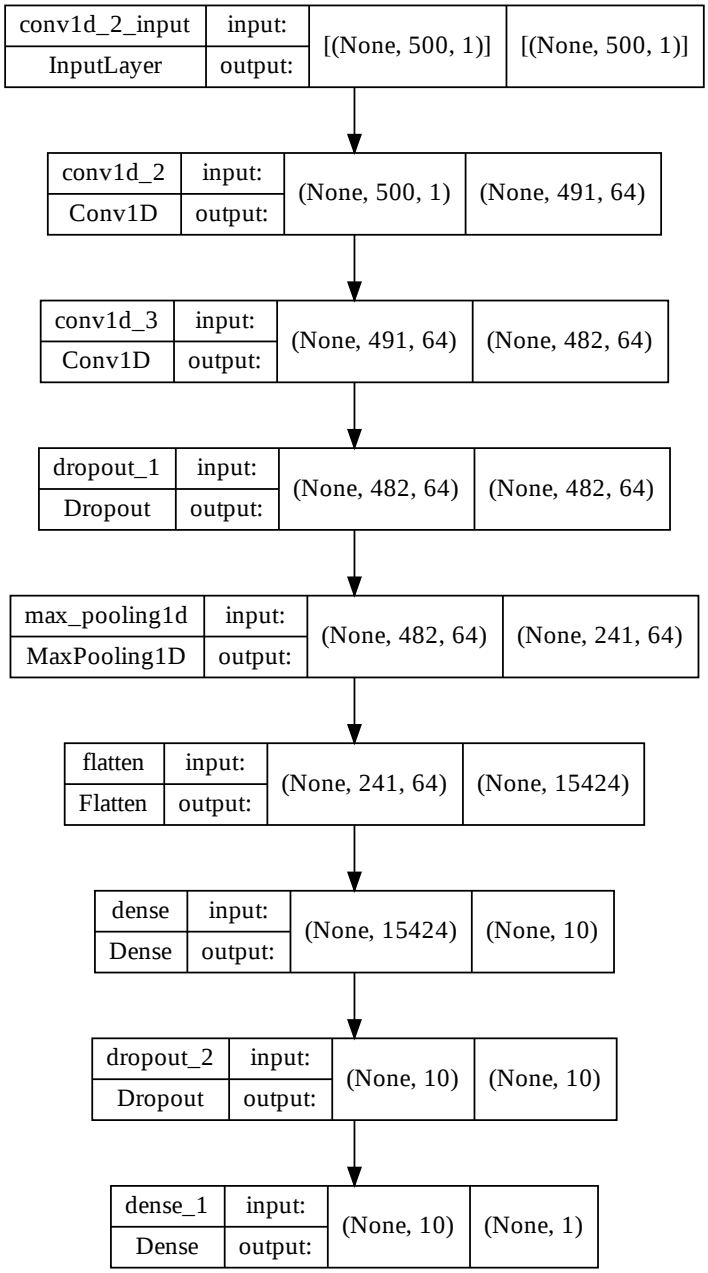


Figure 5.2: Devised CNN structure used for counting underwater heart beats.

Table 5.4: Convolutional layers

Hyperparameter	Setting
Number of Filters	64
Kernel size	10
Strides	1
Padding	Same
Activation	ReLu
Kernel constraint (max norm)	3
Bias constraint (max norm)	3

Table 5.5: Dense layer

Hyperparameter	Setting
Units	10
Activation	ReLu
Kernel constraint (max norm)	3
Bias constraint (max norm)	3

RNN

The structure of the RNN model is detailed by figure 5.3. As shown two recurrent layers, the specific hyperparameter configurations of which are detailed by tables 5.6 and 5.7 respectively. A dropout layer (dropout: 10%) is used after each recurrent layer.

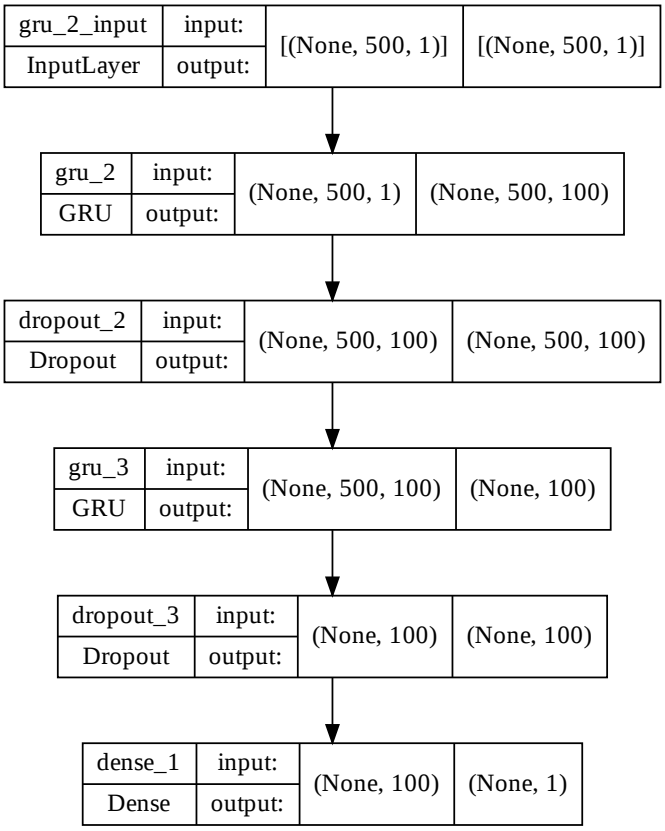


Figure 5.3: Devised RNN structure used for counting underwater heart beats.

Table 5.6: First Recurrent layer

Hyperparameter	Setting
Units	100
Activation	Tanh
Recurrent constraint (max norm)	3
Bias constraint (max norm)	3
Return sequences	True

Table 5.7: Second Recurrent layer

Hyperparameter	Setting
Units	100
Activation	Tanh
Recurrent constraint (max norm)	3
Bias constraint (max norm)	3
Return sequences	False

CNN-RNN

The structure of the CNN-RNN model is detailed by figure 5.4. As shown a convolutional layer followed by a reccurent layer is used the specific configurations of which are detailed by table 5.8 and 5.9 respectively. A dropout layer (dropout: 10%) is used after both the convoluational and recurrent layers.

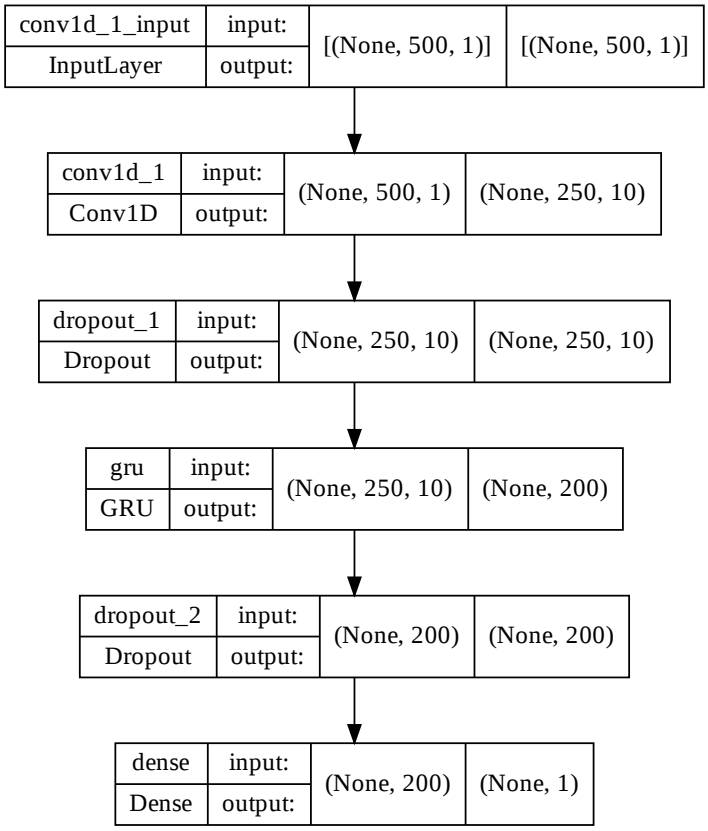


Figure 5.4: Devised CNN-RNN structure used for counting underwater heart beats.

5.3 Results and discussion

5.3.1 Underwater ECG recordings

Initially, it was deemed crucial to understand the noise/distortions observed during underwater ECG sampling. Figure 5.5 demonstrates some examples of the ECG data recorded by the sensor system and pre-processed as discussed above (normalised & resampled). The

Table 5.8: Convolutional layer

Hyperparameter	Setting
Number of Filters	10
Kernel size	10
Strides	2
Padding	Same
Activation	ReLu
Kernel constraint (max norm)	3
Bias constraint (max norm)	3

Table 5.9: Recurrent layer

Hyperparameter	Setting
Units	150
Activation	Tanh
Recurrent constraint (max norm)	3
Bias constraint (max norm)	3

plots are labelled according to which activity was performed during sampling where: (a) is land based; (b) is underwater stationary; (c) is underwater front crawl swimming and (d) is underwater breast stroke swimming.

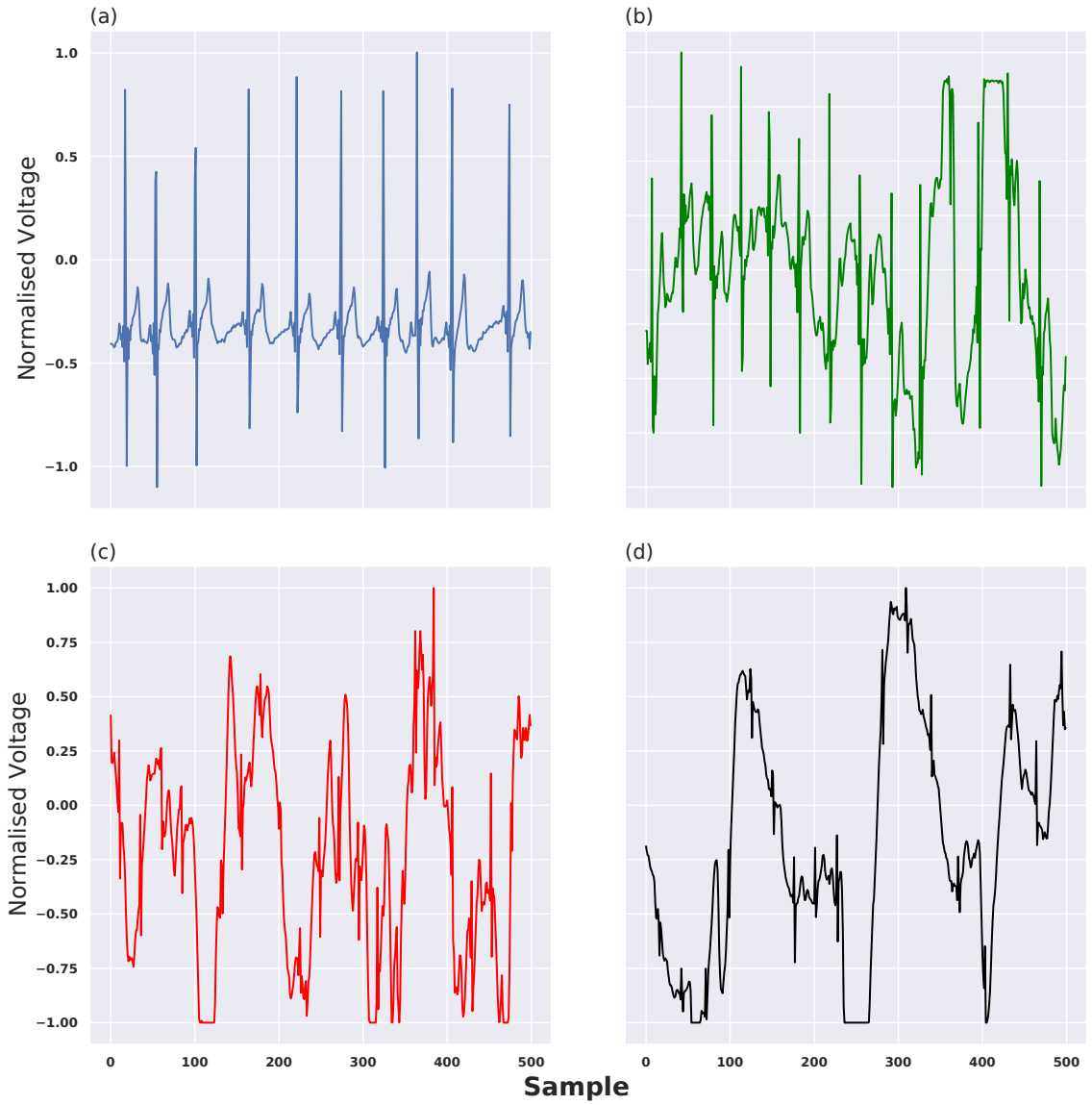


Figure 5.5: ECG samples colour coded according to sampling activity where: (a) is land based; (b) is underwater stationary; (c) is underwater front crawl swimming and (d) is underwater breast stroke swimming.

Crucially, for the underwater samples, the R peaks are generally identifiable by eye despite the signal noise/artefacts. Analysis of the distortions, particularly for the swimming

examples, can correlate the motion artefacts to the type of swim stroke. ECG motion artefacts are common signal errors caused by subject movement during sampling and appear as baseline wander on the signal [159]. The lower frequency baseline wander component seen on the breast stroke example (d), compared to the front crawl example (c), correlates to the rhythm associated with each stroke. Breast stroke involves a low frequency, vertical motion that is larger in magnitude compared to the higher frequency rolling movement associated with front crawl; this is reflected by the two examples shown. The more subtle baseline wander seen on the stationary underwater sample (b) is associated with gentler movements such as body adjustments whilst submerged. The signal saturations seen on all underwater examples (b, c and d), appear as noise when the ECG amplifier operational capabilities are exceeded. This could be caused by sharp body movements causing the amplifier to saturate potentially covering important signal information. Signal saturation is particularly detrimental especially when labeling the samples considering saturation completely obscures the signal. Therefore, in certain cases, the R peaks are not visible by eye. In such situations, the position of the R peak(s) must be estimated by evaluating the remainder of the un saturated sample and predicting where R peaks occur. As per figure 5.5, periods of saturation are short hence this approach was deemed an acceptable method for labelling samples. In addition to this, as shall be detailed below, the neural network apporach detailed by this chapter can be trained on saturated signals with the true beat count value known considering the replification of artefacts. Another effect noticed on the underwater samples is that R peaks may not have a consistent magnitude across the sample. The precise cause of this may be the result of multiple factors, importantly it does not appear to fully obscure R peaks.

The subsequent sections shall discuss the application and performance of all discussed methods (CWT, Kalman Filter and neural networks). To evaluate each approach the MAE

and RMSE are used to indicate how well each technique can count heart beats from noisy 10s underwater ECG samples.

5.3.2 Peak Detection

Application of peak detection (configured as per section 5.2.2) to the 525 underwater ECG samples gave the results demonstrated by table 5.10. This simple approach was not expected to perform well and this is reflected by these results.

Table 5.10: Peak detection performance on underwater ECG samples.

Metric	Score
MAE	4.865
RMSE	6.368

5.3.3 CWT

Application of CWT (configured as per section 5.2.3) to the 525 underwater ECG samples gave the results demonstrated by table 5.11. This technique demonstrates large error and is not suitable for for this task. This is clearly reflected by comparing these results to those of peak detection (table 5.10). Which demonstrates CWT achieves worse performance compared to the simpler peak detection method. This is despite matching the CWT configuration settings used by MATLAB to locate R peaks from noisy ECG signals. A possible cause for this poor performance is that underwater ECG waveforms are distorted so that the sym4 wavelet no longer adequetly matches the shape of the QRS complex.

Table 5.11: CWT performance on underwater ECG samples.

Metric	Score
MAE	5.205
RMSE	6.373

5.3.4 Kalman Filter

Application of a Kalman Filter (configured as per section 5.2.4) to the 525 underwater ECG samples gave the results demonstrated by table 5.12. This method also generally achieves poor results, it does however perform better than peak detection/CWT. This is expected, especially compared to the peak detect method, considering the Kalman Filter fuses the noisy predictions from this approach. Despite this the results remain poor, arguably a simple 1D Kalman Filter is not sophisticated enough to deal with the complex noise/artefacts detailed by figure 5.5. Kalman filters can be investigated in multiple dimensions which was considered as an approach to this problem. This involve estimating the ECG waveform itself as opposed to simply estimating the beat count. Ultimately this approach was not pursued, instead neural networks were investigated. This was done based upon what is discussed in chapter 2, specifically how certain neural network structures (CNN/RNN) are well suited to interpreting sequential data.

Table 5.12: Kalman Filter performance on underwater ECG samples.

Metric	Score
MAE	4.841
RMSE	5.996

5.3.5 Pan Tompkins Algorithm

Application of the Pan Tompkins algorithm (configured as per section 5.2.5) to the 525 underwater ECG samples gave the results demonstrated by table 5.13. This method achieves better MAE results compared to previously applied techniques, however significant RMSE. This indicates that whilst on average residuals are smaller, large residuals remain. Given that this method, a widely used technique for locating R peaks, and all previously demonstrated signal processing/estimation methods perform poorly in this case, it was clear that a more complex solution was required. Hence, neural networks were investigated.

Table 5.13: Pan Tompkins Algorithm performance on underwater ECG samples.

Metric	Score
MAE	2.492
RMSE	7.809

5.3.6 Neural Networks

The approach taken for applying the neural networks is very different compared to that of the previous methods. Neural networks are significantly more complex and cannot be so simply applied as is the case for peak detect, CWT or the Kalman Filter. Principally, neural networks require training data from which to learn in order to make predictions. Often a large volume is required so to ensure network performance and to minimise overfitting [160]. For the underwater application described in this chapter, collecting a large volume of underwater data for training is difficult considering the requirement for pool access, particularly when acquiring swimming samples. Hence land based ECG samples are used for training/validation and so 7359, 10 second ECG samples have been collected. To generate a range of BPM values from this ECG data, various activities such as: sitting;

ascending stairs or jogging on spot were performed during sampling. Each individual sample is noramlised to a range of (-1,1) and resampled at 50 Hz using MATLABs resample function.

The samples are labeled via methods described in chapter 2 using MATLABs peak detection algorithm [161]. Configuration of the peak detect algorithm is peak height: 0.7 and minimum peak spacing: 0.5. Having labeled this land data set, it is split in a stratified manner into training and validation data subsets with ratio 80:20 respectively. In order for a network trained and validated on such data (clean ECG waveforms) to perform well on underwater ECG data, the signal noise/artefacts seen during underwater sampling must be replicated. This process is detailed in section 5.3.6.

As highlighted, when generating the training data, various activities were performed in order to yield ECG samples with a range of R peak counts. The resultant distribution of labels is positively skewed (features more low R peak count samples) as this data was mostly recorded whilst sitting. A positive skew is not noted for the underwater ECG data set as many of these samples were recorded whilst swimming which generated more increased R peak count samples. To account for this the number of underrepresented beat count samples, in the train and validation data sets, was increased. This was done by segmenting underrepresented samples into 5 sections which were randomly shuffled resulting in a new ECG sample. This shuffled sample has the same number of R peaks as the original, however is a new sequence for the neural network to interpret. Repeating this process allows new samples to be generated for underrepresented data labels, however there are limited permutations possible considering each sample is divided into 5 segments. Hence this process is used sparingly to avoid sample duplication. Crucially, this balancing is performed independently on the train/validation subsets after the data is split.

Data augmentations and artefact replication

To ensure neural network performance, the noise/artefacts highlighted above were replicated by applying various modifications to the initially clean training data. These modifications are demonstrated by figure 5.6. Shown here is an original sample (a) that is modified via multiple approaches. As discussed, for underwater samples, particularly when swimming significant baseline wander is present that is dependent on the activity. Traditionally, baseline wander is removed via use of appropriate signal filtering. However, the intention here is to avoid the use of such techniques. Instead a composite signal, formed by summing 3 randomly generated sin waves, is synthesised which is added to a clean ECG signal. This results in a new sample that has a random baseline wander. An example of this is demonstrated by the sample (b) on figure 5.6. Furthermore, to allow the neural network to interpret the signal saturations/inconsistent magnitudes discussed previously, these artefacts are also replicated. See figure 5.6: (c) - compressed sample; (d) - high frequency, short period saturation sample and (e) - low frequency, long period saturation sample. These are controlled by constrained random variables determining, where appropriate, the length, number or magnitude of modifications to the sample. To increase the data set size, the original sample (a), is reversed and the processes detailed by figure 5.6 are repeated for this reversed sample, further increasing dataset size. Note, these modifications are applied to both the train and validation datasets post split.

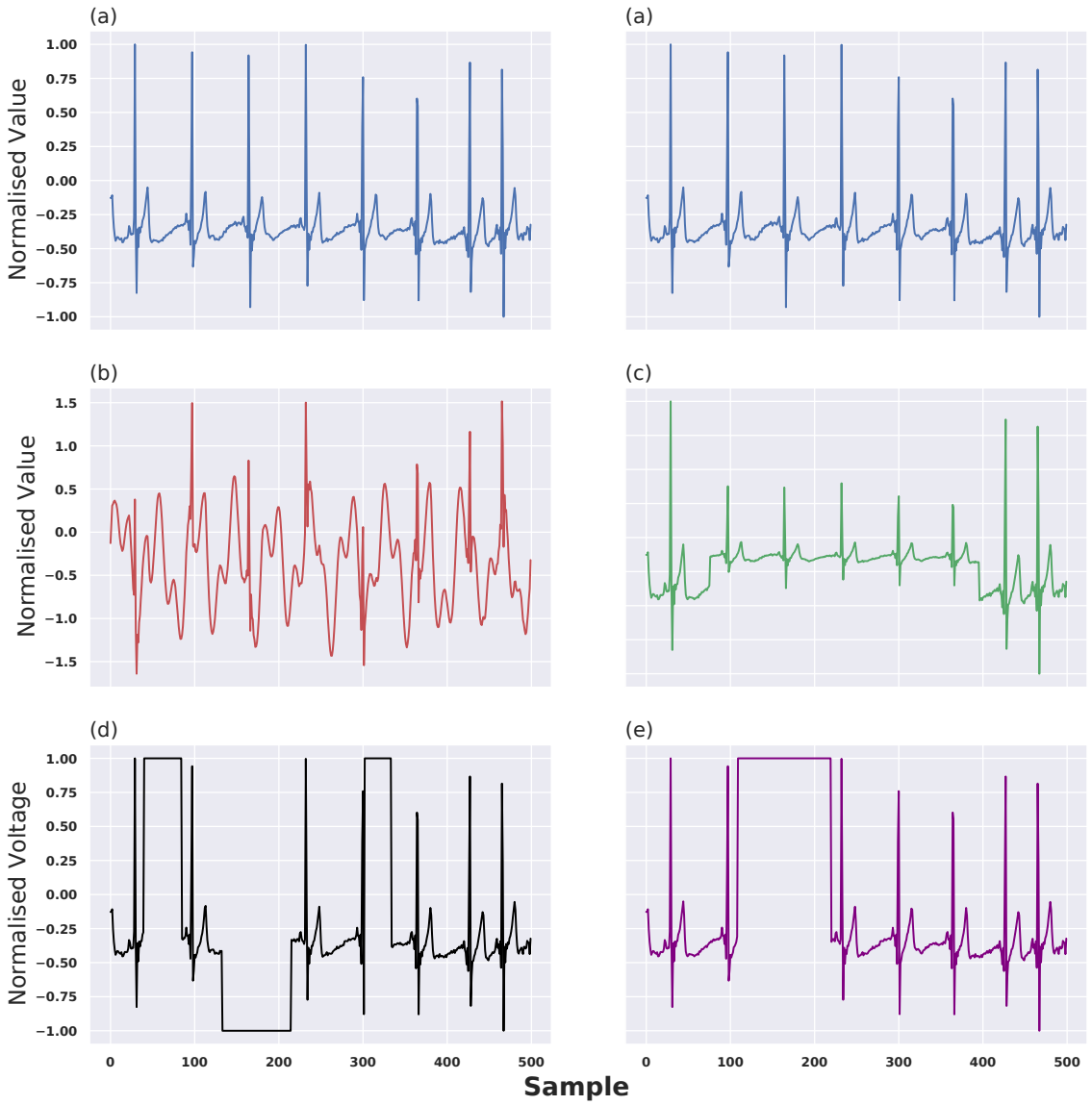


Figure 5.6: Example of how an original ECG sample (a), is modified by: addition of baseline wander (b); compression (c); high frequency, short period saturation (d) and low frequency, long period saturation (e).

Figure 5.7 demonstrates how the modifications detailed by figure 5.6 are applied in combination to an original ECG sample. This process is repeated for all samples in the

training/validation subsets (7359, 10 second land based ECG samples). All samples shown by figure 5.7 are included for training/validation purposes. This approach is used to generate samples that are representative of the underwater ECG data and to augment the size of the training/validation subsets. Augmentation is beneficial because increasing the size of the training set reduces neural network over fitting [160]. Data set augmentation is often used in the field of image classification (commonly performed via use of a CNN [162]), where simple strategies such as the addition of random noise and/or image flipping are used [163]. This generates a modified image that can be used for training as well as the original sample, the modifications detailed by figure 5.6 are applied in combination to replicate this principle.

Figure 5.7 also shows how the combination of modifications can yield samples for which the R peaks are difficult to locate manually. The labels for such samples are known, as the modified samples are associated with un-modified data for which the R peak count has been determined via peak detect. This means effective training on the heavily distorted samples is possible and therefore the devised neural network can attempt to interpret such sequences.

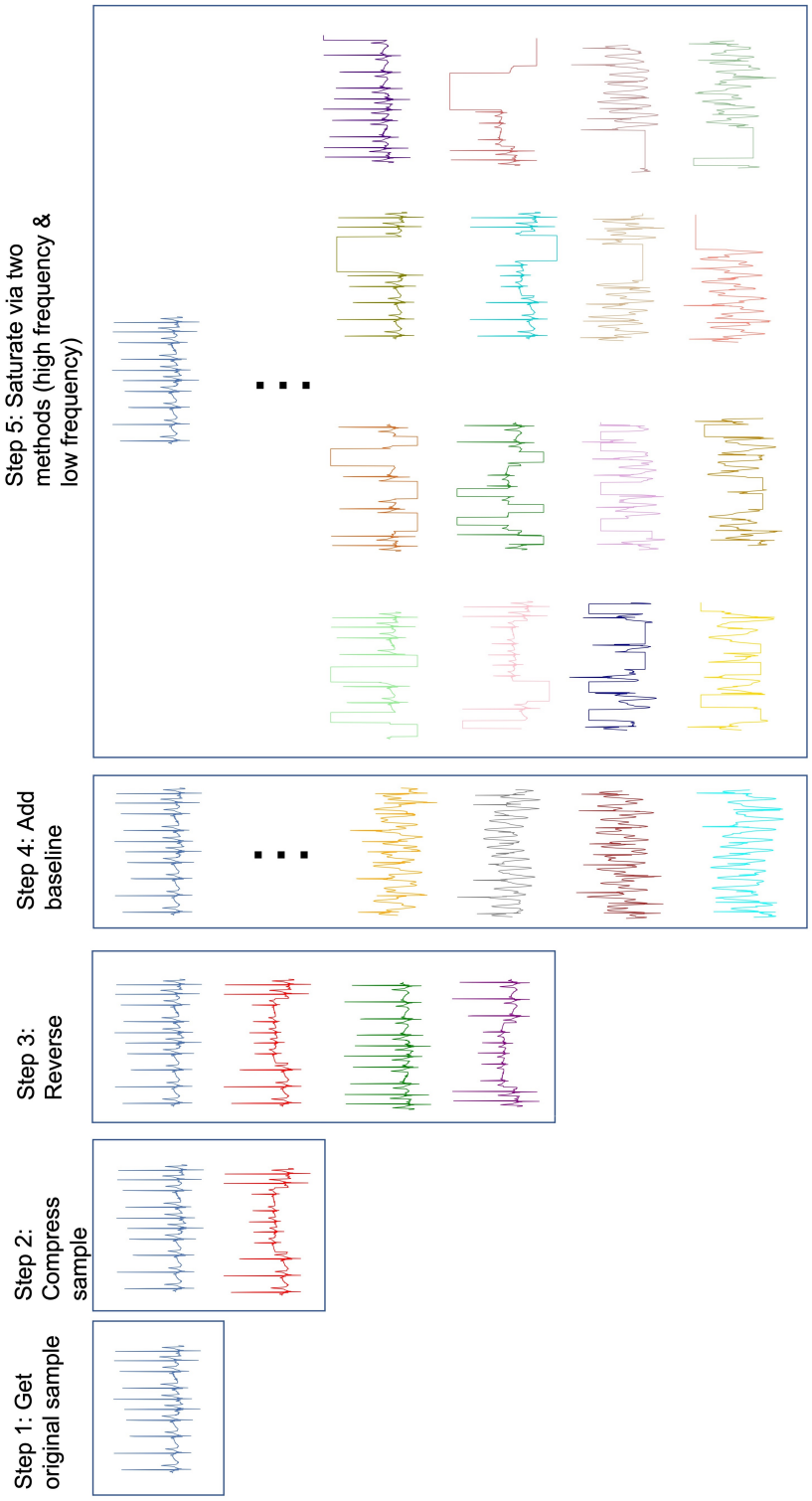


Figure 5.7: Flow chart detailing how an original sample is modified and how the modifications demonstrated by figure 5.6 combine.

Neural network performance

As with previous methods, the three neural network structures are evaluated on the 525 underwater ECG samples. Given the amount of random process associated with generating train/validation sets, it was theorised that the distribution of the random variables was likely to affect each model trained on a different sub sets resulting in a high standard deviation of scores. This is assumed considering the low bias/high variance nature of appropriately tuned neural networks. The randomness associated with artefact generation, in addition to train/validation splitting and weight initialisation, was investigated by training each structure (CNN/RNN/CNN-RNN) ten times on training/validation sets generated per model fitting. Table 5.14 demonstrates high variance of results when tested on the underwater samples confirming the ideas discussed above. To resolve this, ensemble methods are employed in attempt to alleviate the susceptibility to changes in the training/validation sets. A simple averaging strategy is used that calculates the mean prediction of the 10 models trained per dataset which significantly improves results as demonstrated by table 5.15. Also shown is that the CNN-RNN ensemble significantly out performs the other methods (CNN/RNN) and is drastically better than the peak detect, CWT and Kalman Filter approaches.

Table 5.14: Neural network performance on underwater ECG samples.

Model	Mean RMSE	Standard deviation of scores
CNN	1.47	0.13
RNN	1.25	0.18
CNN-RNN	0.96	0.17

Considering the performance of the CNN-RNN ensemble, further metrics are examined as demonstrated by table 5.16. The MAE indicates that on average, this model is less that 0.5 heart beats incorrect compared to the true value and the ME indicates that the model

Table 5.15: Neural network performance on underwater ECG samples.

Model	Ensemble RMSE
CNN	1.13
RNN	0.84
CNN-RNN	0.58

tends to overestimate the BPM value. Although the overestimation is small. In addition the R^2 value indicates the CNN-RNN ensemble is well suited to calculating the number of heart beats from underwater ECG samples.

Table 5.16: Neural network performance on underwater ECG samples.

Metric	Scores
MAE	0.46
R^2	0.99
ME	0.003

To understand the importance of applying the transformations prior to neural network training, 5.17 demonstrates the testing results of a CNN-RNN model (as per figure 5.4 & tables 5.8/5.9) trained on unmodified ECG data when applied to underwater data. As demonstrated, poor performance is noted. In this case the neural network performs worse than the peak detection, CWT and Kalman filter methods highlight the importance of artefact replication. Previous work involving a single neural network (as opposed to the ensemble method discussed here), indicated that the compression artefact in particular was crucial modification to include [2].

Table 5.17: Effect of artefact removal from train/validation sets.

Metric	Scores
MAE	5.782
RMSE	7.881

5.3.7 Results summary

Whilst the use of a neural network in any case is associated with increased computational/mathematical complexity, it can be argued that the results demonstrate the potential of a CNN-RNN approach. Also, minimal data pre processing is required with no need for any signal filtering. A software specific approach to this problem means that no special design is necessary with regards to: the chest strap; the electrodes or the amplifier ensuring such an approach is low cost and easy to distribute. Whilst the application demonstrated in this chapter is relatively simple (counting heart beats during a 10 second sample), the approach employed to make normal ECG samples representative of underwater samples could be used to train a neural network to recognise many other important ECG measurable factors such as Heart Rate variability (HRV).

Future work beyond what is discussed in this chapter will involve continued investigation of the artefact replication methods. As demonstrated by these results, good neural network performance (when tested on underwater ECG data) is achievable for models trained on modified land sampled ECG data. This concept will be applied to data sampled from other sensors, such as breathing rate so to investigate the viability of artefact replication for other psychological metrics. Furthering this idea, applying similar artefact replication methods in other environments where a large training/validation set is difficult to collect is potentially beneficial.

5.4 Conclusions

This chapter investigates several approaches to identify hearts beats from fixed length ECG sequences sampled underwater. This is done as a method to estimate BPM which is an important physiological measurement generally and for applications such as performance sport; rehabilitation and/or divers in hazardous environments. The work demonstrates that a CNN-RNN architecture preforms significantly better compared to other neural networks and BPM generation techniques. This process requires minimal signal pre-processing prior to network application. In addition, no specific electrical/mechanical considerations are necessary. The CNN-RNN network has been trained on land based ECG based which has been modified to represent underwater ECG data. Testing performed on this genuine underwater ECG data shows that the modifications applied yield representative samples suitable for training/validation purposes.

Chapter 6

Underwater Signal Analysis

Chapter 5 analyses the ECG data recorded by the underwater sensor system variant. However, as discussed in chapter 3, the sensor platform is also capable of monitoring breathing rate, acceleration (via triple axis accelerometer) and temperature data which have not yet been included for analysis. This chapter considers the data from multiple sensors (ECG, accelerometer, breathing) and the application of machine learning techniques for swim analysis. This is achieved via the use of the BPM generation methods detailed in chapter 5, with the new algorithms introduced here. Primarily, this chapter demonstrates swim stroke classification models that classify whether a subject is swimming: front crawl; breast stroke; back stroke or is not swimming via data recorded from a chest situated accelerometer. It is crucial that the device be located in this region so that the sensor platform as a whole remains compact (positioned on a chest strap) and comfortable during swimming. As stated in chapter 1, no methods for swim stroke classification from the chest region have been found in literature. In addition to swim stroke classification, methods for estimating swim breathing and stroke rate are also discussed for which reliable swim stroke classification is a necessary prerequisite. The methods detailed in this chapter could

feasibly be adapted for application to other scenarios such as scuba diving or search and rescue scenarios, these ideas are expanded upon below.

Thus far, all techniques demonstrated have been tested at home/in an indoor swimming pool. This has been done to verify performance and to show initial results. In the future, deployment into open water settings warrants investigation of the LoRa protocol for data transfer (instead of WiFi), which would significantly increase the range of transmission. This adaptation would be beneficial as it would allow for remote monitoring capabilities where swim trainers may want to assess performance at a distance. Adopting LoRa could also be beneficial to search and rescue scenarios whereby the decision to maintain search efforts in dangerous conditions, at sea for example, may be motivated by the identification of a subjects BPM/breathing rate/ability to swim. With this, RF direction finding methods could also be integrated for further assisting search and rescue applications. Depending on the situation, this may then require investigation into the integration of: chest electrodes; ECG amplifier and radio transmitter into appropriate clothing. Which would be required if the subject in question were to unexpectedly enter the water, for example, man over board situations. This idea is particularly interesting considering no similar examples of this concept can be found in literature. However, testing in open water environments may require some modification to the devised algorithms, particularly when investigating moving water applications. The sea for example where waves would likely impact accelerometer readings. This shall be further discussed in later sections.

This chapter presents three classical machine learning solutions and a neural network approach to swim stroke classification. Breathing/swim stroke rate estimation techniques are then discussed using a dedicated sensor or via approximation from accelerometer data, for which swim stroke identification is key as will be detailed in subsequent sections. An example of how the previously developed BPM generation algorithms can be used with the

techniques developed in this chapter, is then demonstrated. This is done as an example to show the analytic potential of this work.

6.1 Methods

6.1.1 Data acquisition process

The sensor system variant used to generate the results discussed in this chapter is detailed by chapter 3, section 3.2.3. As discussed the device focuses on accelerometer (all 3 axes), ECG and breathing data. The sensor system, attached to the chest strap, is orientated such that the waterproof enclosure sits centrally on the back of the swimmer (see figure 6.1). All data presented has been obtained from a single user, in accordance with relevant ethical approval processes in place at the time of testing.

This device has been used to initially record 1496, 5 second accelerometer/breathing samples all collected from 1 subject in a swimming pool (chlorine concentration 1.5ppm), this data is used form machine learning training and validation. The sample length (5 seconds) is chosen to provide regular swim stroke classifications that can be interpreted and used for swimming analysis. A subsequent 554 samples, sampled from 4 subjects (including the initial subject), are taken in identical conditions (from a swimming pool with chlorine concentration of 1.5ppm and 5 seconds in length). These additional data sets are used for machine learning model testing. Training and test samples have been recorded and labelled accordingly by assigning an integer to each swim stroke. Generally, during a period of recording, one swim stroke was swum so that all samples recorded could be labeled as the given swim stroke. However, some manual labelling was necessary particularly for 'not swimming' samples. All data is recorded onto an SD card and analysed post swim. This approach is selected instead of WiFi transmission, as used in chapter 5, as

data transfer via wireless methods was found to be unreliable during back stroke swimming during which WiFi transmission fails. This lapse in performance can be related to chapter 4, where it was found the increasing submission depth, as backstroke does considering the sensor system orientation during swimming (see figure 6.1), significantly impacts the RSSI.

6.2 Results and discussion

6.2.1 Underwater accelerometer/breathing recordings

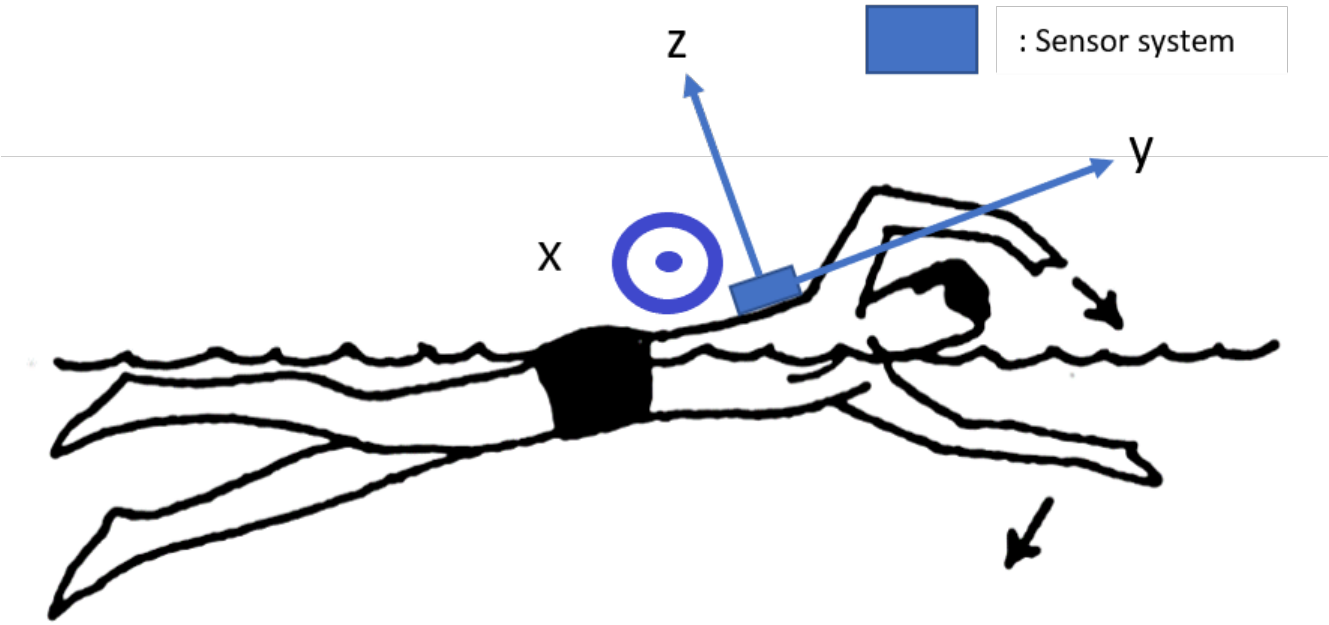


Figure 6.1: Diagram detailing orientation of sensor system, with accelerometer axes labelled. Note, that although the x axis is not perpendicular to the y/z axis for this representation, this has been done to make the plot clearer. In reality, the x axis is perpendicular to the y/z axes.

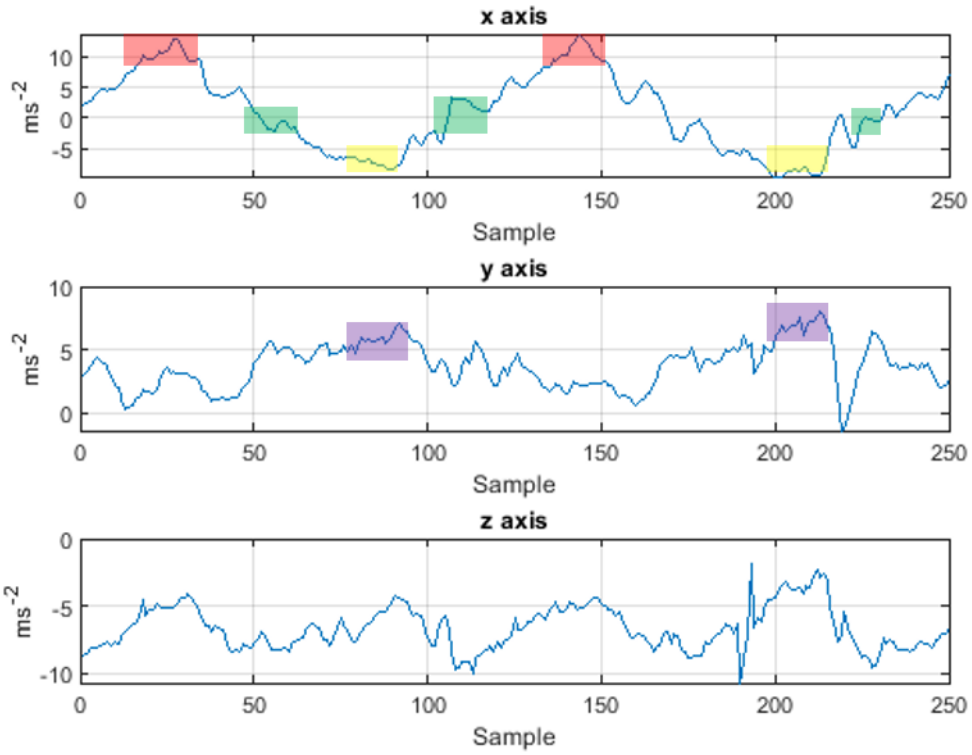
Similar to figure 3.5, figure. 6.1 details the orientation of the sensor system relative to the swimmer. The sensor system is worn in this orientation for all swim strokes considered

in this chapter. However, in this case the accelerometer axes have been labeled. These labels indicate the positive direction for each axis, the x axis is not labeled as it directs out of the page. Figures 6.2 - 6.4 demonstrate examples of accelerometer and breathing data recorded by the sensor system during front crawl, back stroke and breast stroke swimming respectively. A 'not swimming' acceleroemeter sample is not demonstrated considering there is no specific signal features to analyse. Correlating the accelerometer samples with the breathing samples, provides crucial insight into the data and demonstrates that the movements associated with each stroke could be identified from the recorded data. This is expanded upon below. It should be noted that the accelerometer axis values recorded during each swim stroke are specific to the orientation of the accelerometer sensor within the enclosure. Exact replication of the subsequent accelerometer data requires matching the orientation to that shown by figure 6.1.

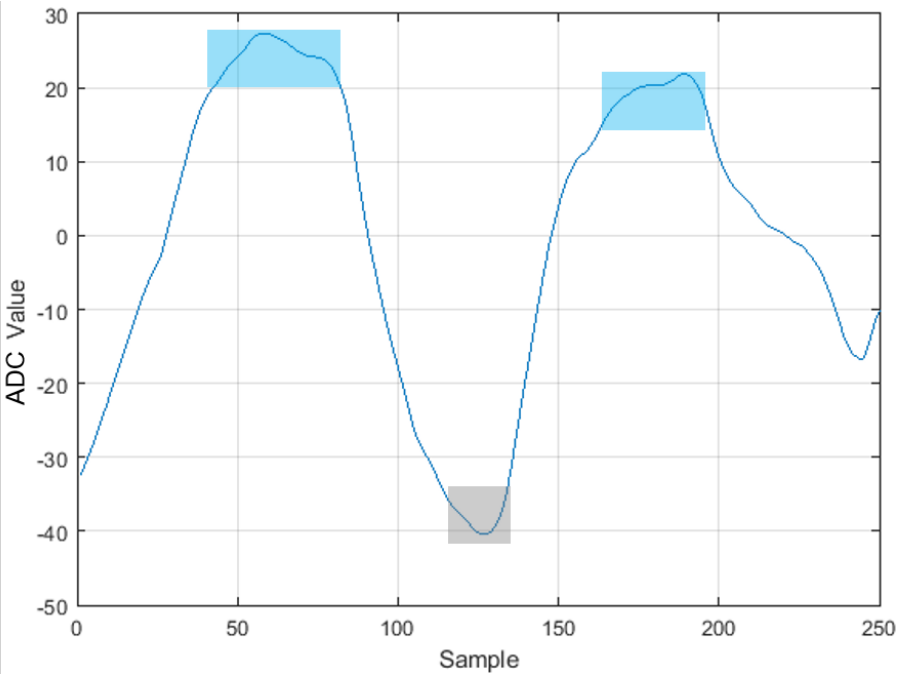
Front Crawl

Figure 6.21 demonstrates an example of the x, y, z accelerometer axis data recorded during front crawl. For this example, front crawl is swum so that a breath is taken every two strokes, this means breathing is always performed on the same side. The x axis clearly demonstrates the rolling motion associated with the swim stroke, which involves rotating the chest to breathe sideways as the arms rotate vertically through the water and air. As the body rotates round for a breath, the accelerometer readings on the x axis increase. The x axis values peak when the body achieves maximum rotation out of the water (shaded red on the accelerometer y axis). This is when the maximum component of acceleration due to gravity acts, on the x axis. Inversely, the x axis readings decrease as the body rolls back round into the water (body is briefly level when x axis measures approximately 0 ms^{-2} - shaded green on the x axis) and the next stroke begins. The head remains submerged for

the next stroke with the opposite arm being raised out of the water. As demonstrated, the same magnitude of acceleration is not achieved on this counter roll (shaded yellow on x axis) which is logical considering the body does not fully rotate out of the water on this side. Cross referencing this to the corresponding breathing data (figure 6.21), breaths (chest expanding - maintained increase in ADC values) are demonstrated matching this rolling motion. Naturally, the breaths lag the rolling motion of the body as the mouth has to rotate fully out of the water prior to a breath being taken. As demonstrated, maximum chest expansion (peaks of figure 6.21 - shaded blue on the breathing graph) occurs approximately when the body in level in the water prior to the swimmers face going back underwater. As the swimmer exhales underwater, the chest contracts to its original dimensions (shaded grey). The y axis demonstrates that generally the swimmer achieves higher forward (in direction of swimming) acceleration (increased y axis values, shaded purple on accelerometer z axis) when the body rolls back into the water (decreased x axis values). This is likely the result of multiple factors associated with front crawl swimming, one possible reason being a reduction in drag when the swimmers head returns to a more streamlined position through the water post breath. The z axis also demonstrates a sinusoidal rhythm caused by the rolling nature of front crawl swimming. Similar to the x axis, as the z axis rotates, the component of gravity effective on it changes which is reflected by the data. Crucially, the z axis is dominated by a largely negative acceleration which is caused by the acceleration due to gravity.



(1)



(2)

Figure 6.2: 6.21 accelerometer and 6.22 breathing data recorded during front crawl swimming

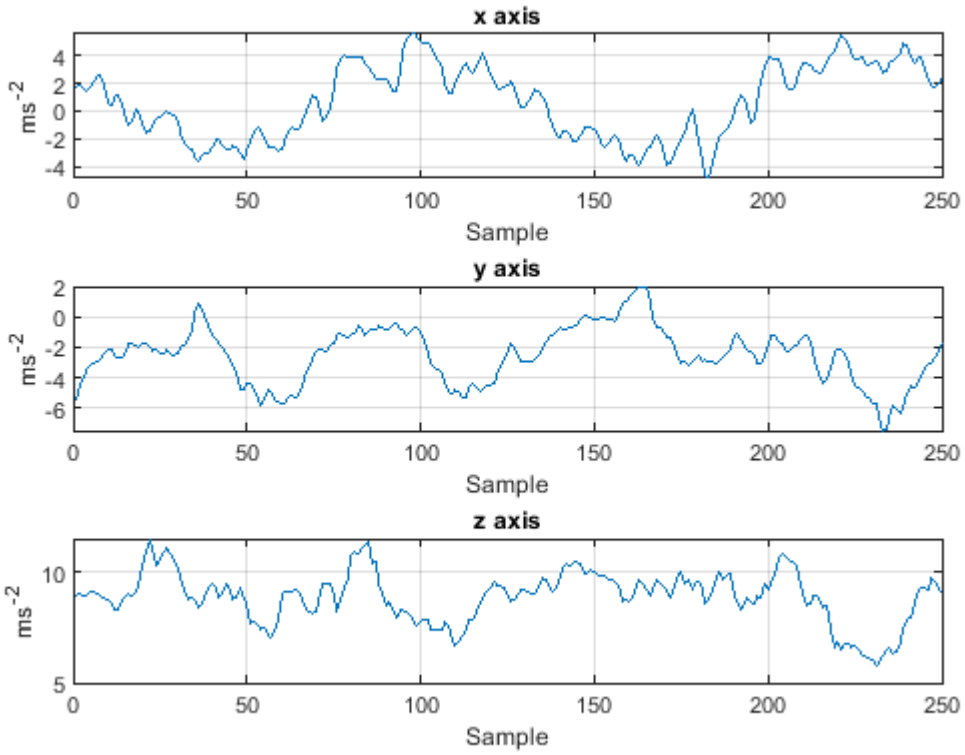
Back Stroke

Figure 6.31 demonstrates the x, y, z accelerometer axis data recorded during back stroke swimming. As shown back stroke demonstrates a similar x axis rolling motion, compared to figure 6.21, as the arms rotate and propel the swimmer forward. For back stroke, the breathing rate is less dependent on the rolling motion considering the swimmers mouth is not submerged. However, it is possible that the subject may naturally sync the rotation of a particular arm with taking a breath as appears to be the case in figure 6.32. The range of ADC values recorded during back stroke (and subsequently breast stroke) is significantly smaller compared to that of front crawl. Indicating the magnitude of chest is expansion is greater during front crawl, possibly because it is a more energetic stroke. The y axis also demonstrates a sinusoidal pattern relating, as with front crawl, to the changes in swimming forward acceleration. However in this case the values are all negative. This is likely the case due to the angle of the waterproof enclosure relative to the swimmers torso, see figure 6.1. In addition, during back stroke, swimmers often have an arched back which results in a significant component of acceleration due to gravity acting on the y axis. The net effect being that average y axis acceleration is negative, however this does not mean that the swimmer is primarily decelerating. Like front crawl, the z axis is also dominated by g. However, unlike front crawl, the direction of this acceleration is reversed, which is as expected considering the physical orientation of the sensor during these two swim strokes.

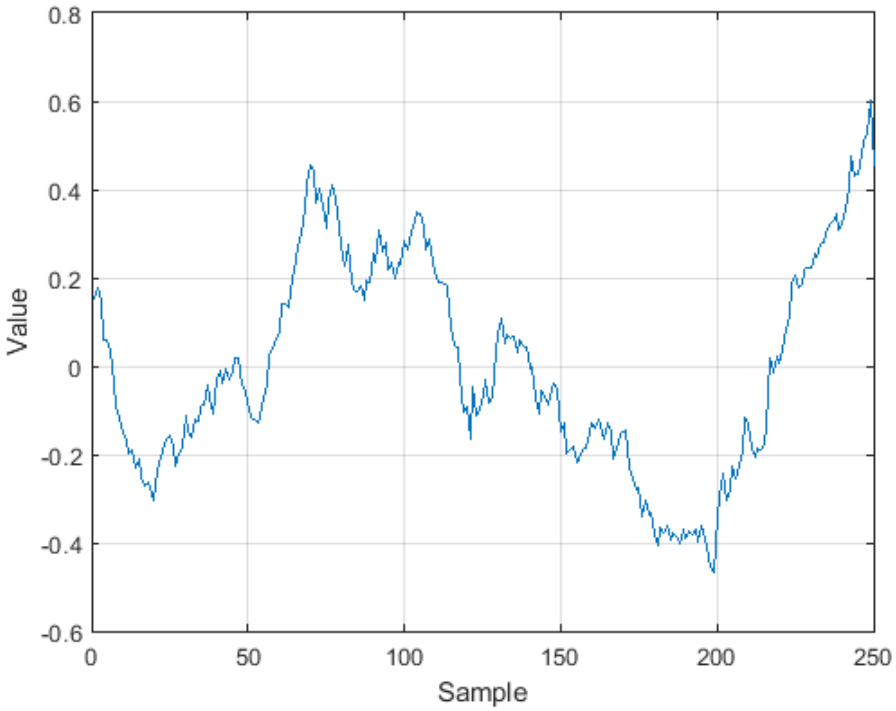
Breast Stroke

Figure 6.41 demonstrates the x, y, z accelerometer axis data recorded during breast stroke swimming. As demonstrated, breast stroke exhibits significantly different accelerometer patterns compared to front crawl (figure 6.21) and back stroke (figure 6.21); which is expected considering the movements associated with each of the three swim strokes in-

vestigated. There appears to be some correlation between the swimmer taking a breath (maximum chest expansion - regions shaded blue on the breathing graph) and increased y axis values (indicating an increase in acceleration in the direction of swimming - regions shaded purple on accelerometer y axis) and z axis (indicating the swimmers torso is accelerating upwards - regions shaded red on accelerometer z axis) acceleration values. Breast stroke involves taking a breath by raising the body out of the water, after which the arms rotate in a horizontal circular motion to propel the swimmer forward. It is therefore logical that breathing should correlate to and increase in y and z axis acceleration. There is also an increase in y acceleration after a breath is taken (regions shaded green on accelerometer y axis). This is likely due to the leg kick associated with breast stroke post breathing. The x axis is not thought to be significant for breast stroke considering the relatively flat horizontal profile of the swim stroke. The signal changes demonstrated by figure 6.41 compared to figures 6.21 and 6.31 are smaller in magnitude, it was thought that this would make breast stroke harder to identify automatically. However, the subsequent results do not indicate this.

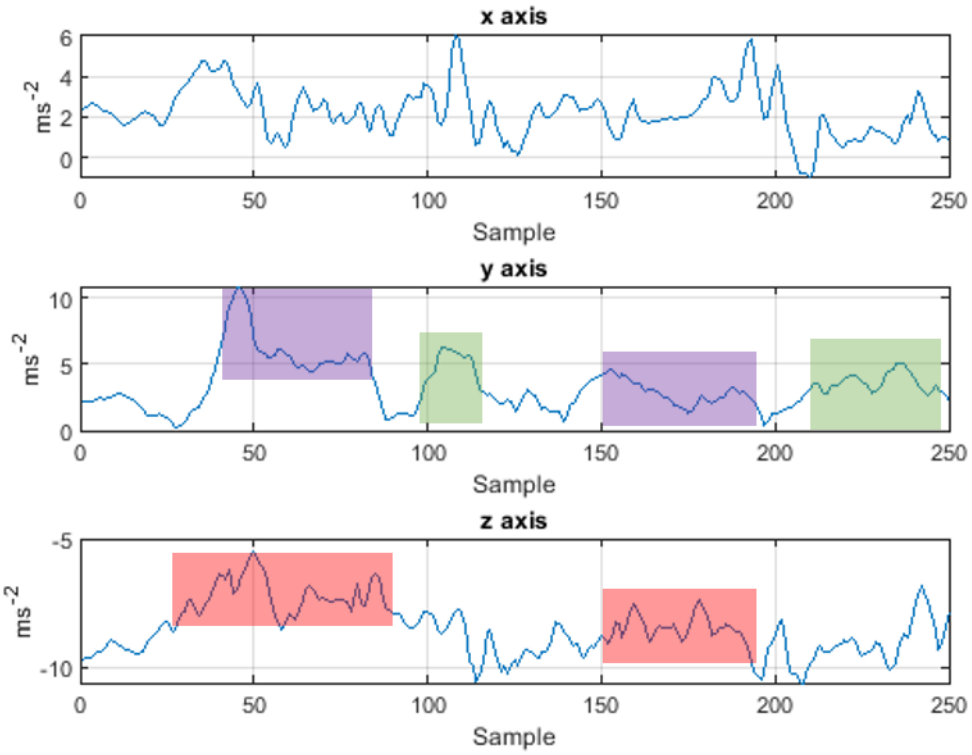


(1)

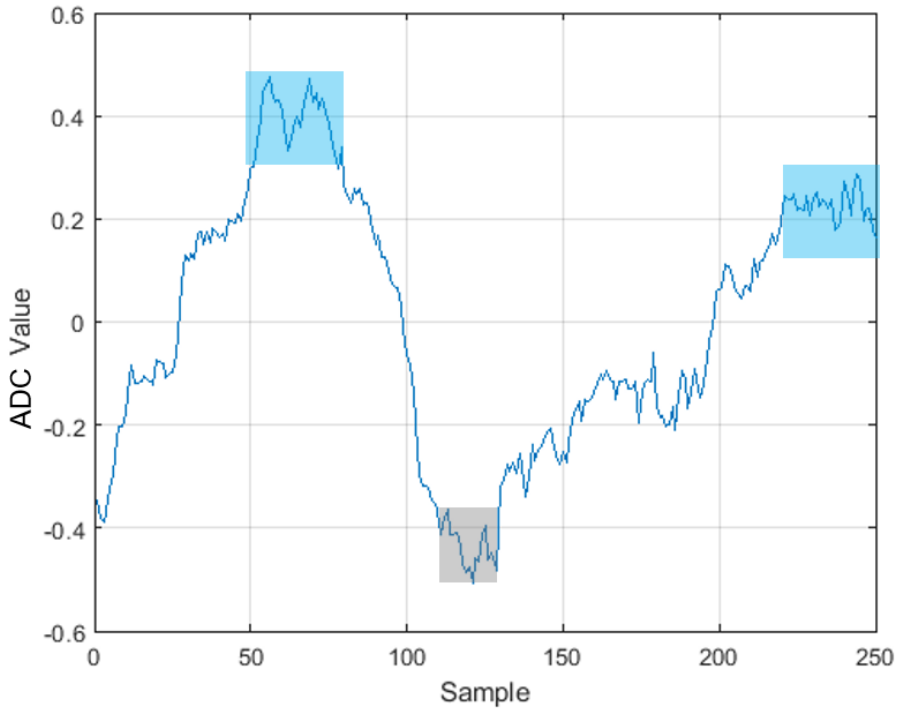


(2)

Figure 6.3: 6.31 accelerometer and 6.32 breathing data recorded during back stroke swimming



(1)



(2)

Figure 6.4: 6.41 accelerometer and 6.42 breathing data recorded during breast stroke swimming

To classify swim stroke, only the accelerometer data is used. Initially neural network approaches (in particular CNNs/RNNs) were considered due to the use of sequence data (accelerometer samples) as with chapter 5. However, as discussed previously CNN/RNN structures are computationally expensive methods that often require a large training dataset in order to be applied effectively. Hence other, machine learning approaches were considered. The issue is that the algorithms investigated here (KMeans, KNN, SVMs, dense neural networks etc) are not usually associated with sequence interpretation as with RNNs or 1D CNNs. The subsequent section explains how features are generated, from the acclerometer sequences, to yield tabular data more suited to simpler algorithms. As discussed, this is done in attempt to classify swim stroke from 5 second acclerometer (x, y, z) samples.

6.2.2 Feature generation

Several features from the accelerometer samples are generated. These features are calculated either from the original signals or from the signals processed via FFT. These two representations were considered because of the motion and frequency of motion associated with each swim stroke.

- Data Range - Sample divided into sub samples. The difference between max and mean is calculated per sub sample. The mean of sub sample ranges is then calculated.
- Mean Delta - The numerical difference (absolute value) between consecutive samples is calculated. The mean of these differences is then calculated.
- Mean - Mean of the sample is calculated.
- Max - Max of the sample is calculated.
- Kurtosis - Kurtosis of the sample is calculated.

- Skew - Skew of the sample is calculate.
- Inter Qualtile Range (IQR) - IQR of the sample is calculated.
- Standard Deviation - Standard deviation of the sample is calculated.
- Frequency Mean - FFT is applied to the sample the mean of which is calculated.
- Frequency Max - FFT is applied to the sample the max of which is calculated.
- Frequency Kurtosis - FFT is applied to the sample the kurtosis of which is calculated.
- Frequency Skew - FFT is applied to the sample the skew of which is calculated.
- Frequency IQR - FFT is applied to the sample the IQR of which is calculated.
- Frequency Standard Deviation- FFT is applied to the sample the standard deviation of which is calculated.

When using FFT, as per chapter 2, 2.1.2, spectral leakage is a common issue. To mitigate spectral leakage, for this work, FFT is configured with 2000 points to ensure a small frequency resolution and a rectangular window is used. Whilst these factors may not completely eradicate spectral leakage, the good results demonstrated by section 6.2.4 indicate the effects are not significantly detrimental. The features generated have been used considering all are widely used statistical methods. No specific selection process was performed, focus was placed on tuning the model during the validation stage to achieve good performance, as demonstrated below. Figure 6.5 demonstrates the correlations between the generated features. As shown, some strong linear correlations exist between these features indicating that some good classification results should be achievable with appropriate model selection.

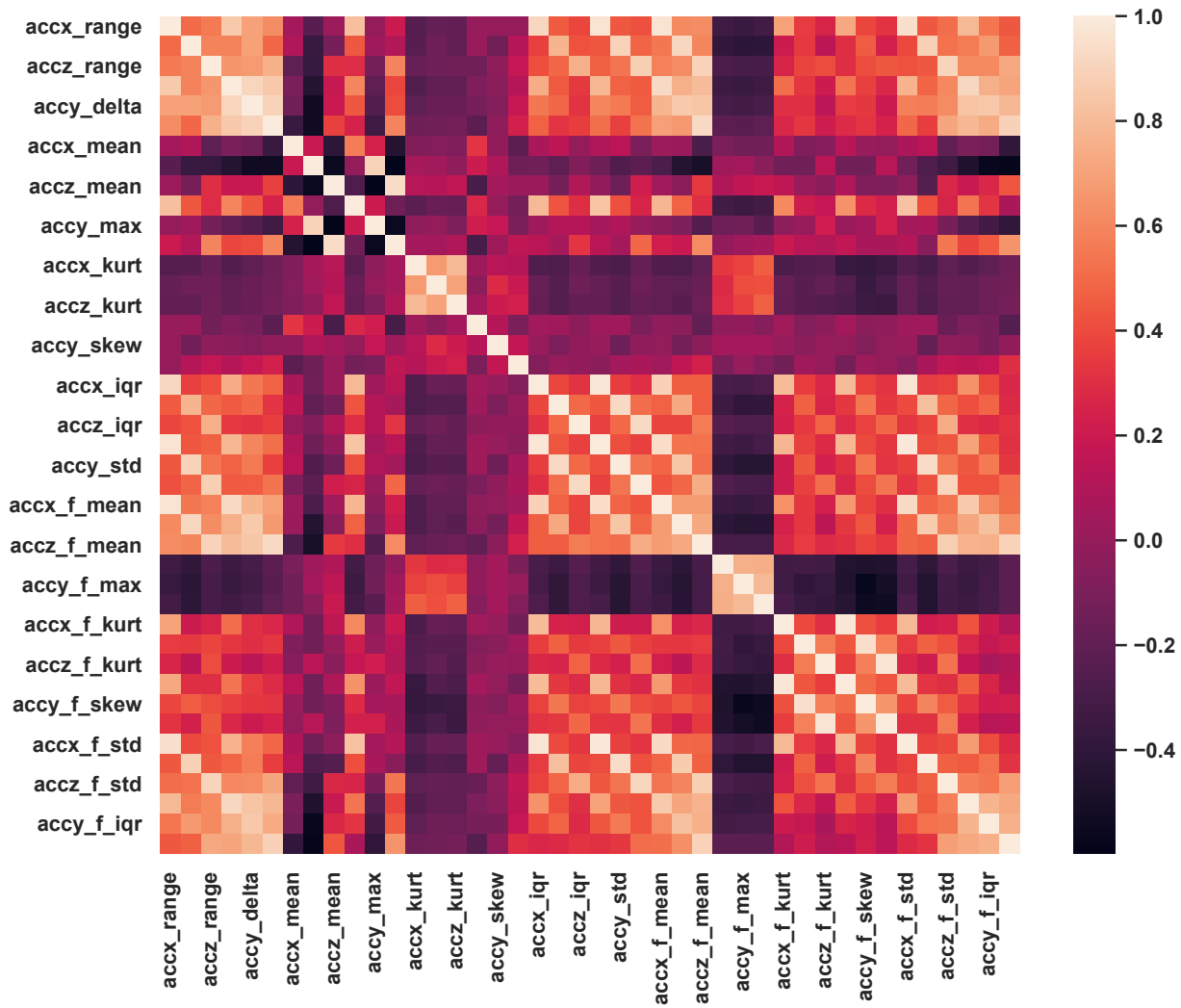


Figure 6.5: Heatmap demonstrating the correlations between features generated from the acclerometer samples.

6.2.3 Unsupervised approach

Initially unsupervised machine learning techniques were investigated, post feature generation the resultant dataset has 42 features (3 accelerometer dimensions multiplied by 14 feature generation methods). As with chapter 7, PCA (see chapter 2, section 2.3.1) was ap-

plied to this data (generated features). Figure 6.6 demonstrates two principle components (which explain 62% of the total variance) colour coded according to the corresponding swim stroke. As demonstrated, for this representation there is significant overlap between 'breast stroke' and 'not swimming'. Despite this, KMeans (see chapter 2, section 2.3.1) clustering (number of clusters set 4) has been applied as demonstrated graphically by figure 6.7. For simplicity, the KMeans clusters are colour coded to match the swim stroke labels demonstrated by figure 6.6. As shown by figure 6.7, KMeans cannot differentiate the front crawl and back stroke data instances. In addition, the not swimming and breast stroke data points are not properly identified as is demonstrated by clusters 2, 3 and 4.

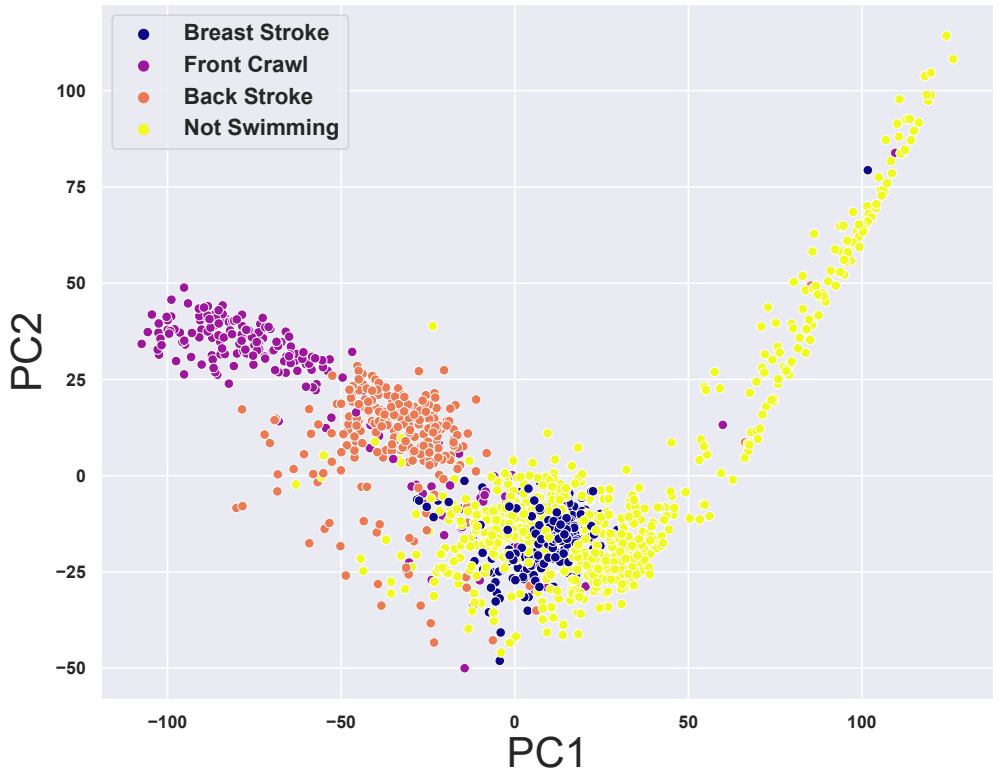


Figure 6.6: PCA in 2 dimensions applied to multi dimension generated feature dataset. Colour coding applied according to swim stroke labels.

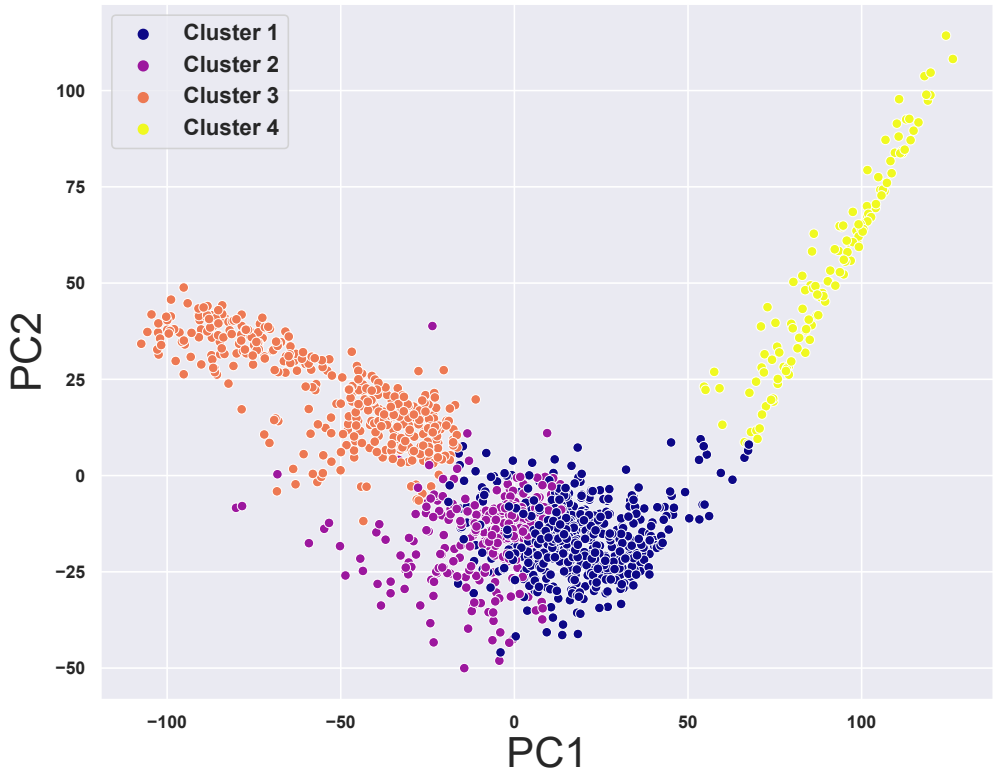


Figure 6.7: KMeans clustering (number of clusters set as 4) applied to PCA processed dataset demonstrated by 6.6

The performance of KMeans applied to the PCA processed data is demonstrated by table 6.1; which shows the classification report between cluster labels and swim stroke labels. Important to note here is the class imbalance within this data set. There are: 352 breast stroke samples; 208 front crawl samples; 262 back stroke samples and 674 not swimming samples samples hence 1496 samples in total. Regardless of the imbalance, KMeans performs poorly as expected considering the lack of separation demonstrated by figure 6.6.

However, further investigation found that if the 'not swimming' data instances are removed, significantly better KMeans performance can be achieved. Figure 6.8 demonstrates

Table 6.1: KMeans clustering performance compared to swim stroke labels.

	Precision	Recall	F1-score	Count
Breast stroke	0.48	0.93	0.63	352
Front crawl	0.43	0.81	0.56	208
Back stroke	0.02	0.01	0.01	262
Not swimming	0.78	0.34	0.47	674
Macro average	0.43	0.52	0.42	1496
Weighted average	0.53	0.49	0.44	1496

the first two principal components, of the same dataset demonstrated by figure 6.6, with the 'not swimming' data points removed prior to application of PCA. Clearer separation is visible having taken this step and application of KMeans (K set as 3) gives much improved results, see figure 6.9. In this case, the identified clusters correspond well to the swim strokes, as demonstrated by table 6.2; which again demonstrates the classification report. Clearly, the performance here is significantly better compared to the results shown by table 6.1.

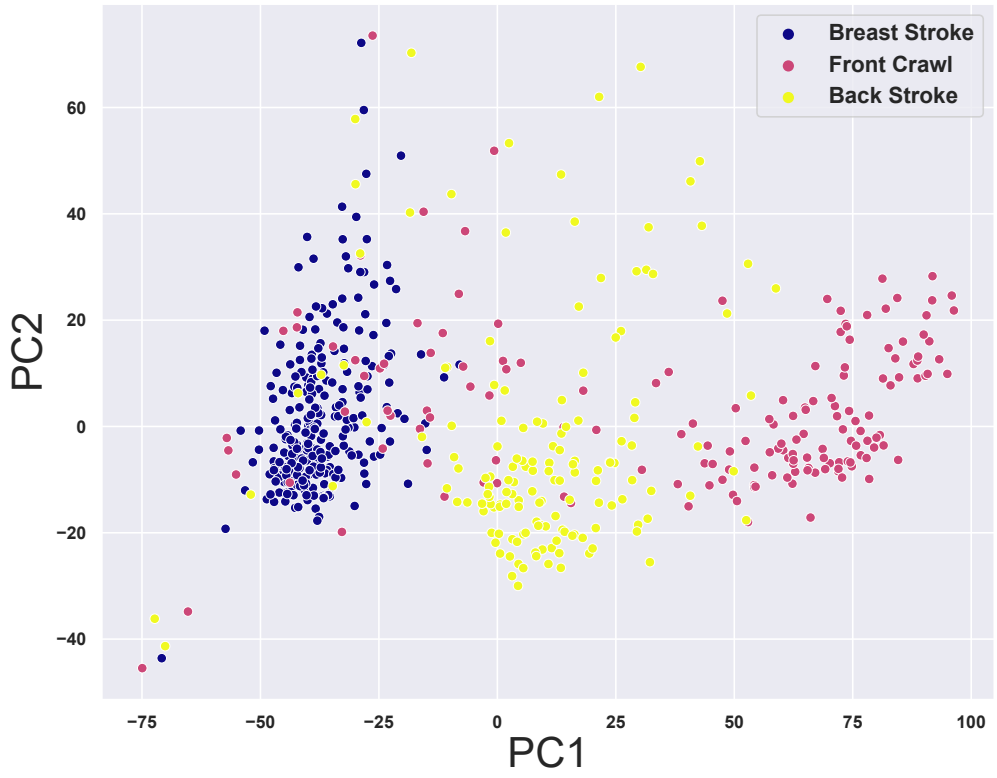


Figure 6.8: PCA in 2 dimensions applied to multi dimension generated feature dataset. 'Not swimming' instances removed prior to application of PCA. Colour coding applied according to swim stroke labels.

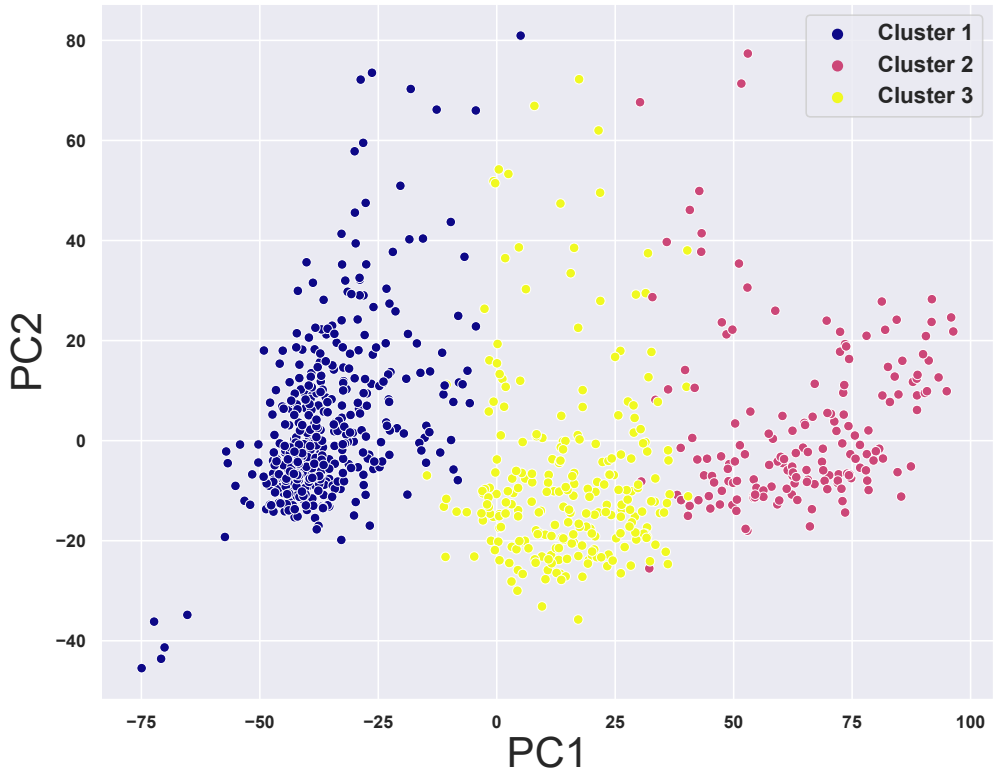


Figure 6.9: KMeans clustering (number of clusters set as 3) applied to PCA processed dataset demonstrated by 6.8

Table 6.2: KMeans clustering performance compared to swim stroke labels, 'not swimming' instances removed prior to application of PCA.

	Precision	Recall	F1-score	Count
Breast stroke	0.86	0.99	0.92	352
Front crawl	0.91	0.74	0.81	208
Back stroke	0.91	0.85	0.88	262
Macro average	0.89	0.86	0.87	822
Weighted average	0.89	0.88	0.88	822

To improve swim stroke classification performance, supervised machine learning algorithms were considered using the swim stroke labels available.

6.2.4 Supervised approaches

Model training

Several classic machine learning models (RFC - chapter 2, section 2.3.2, KNN - chapter 2, section 2.3.2, SVC - chapter 2, section 2.3.2, LDA - chapter 2, section 2.3.2) were investigated via repeated stratified K fold cross-validation (see chapter 2, section 2.2). This method of model evaluation is chosen considering the limited training set size, to estimate a model(s) skill and to provide a standard deviation of repeated scores. The cross-validation process is configured with k and number of repeats set as 10. The metric used for scoring is F1-score (sec chapter 2, section 2.2) weighted average. The weighted average indicates the average F1-score relative to the representation of each class. Table 6.3 demonstrates the three best performing models, with regards to swim stroke classification, on all 4 stroke types. It should be noted, that the results demonstrated are from un-tuned models, yet as shown all outperform KMeans. As per table 6.2, at best KNN achieves weighted F1 score: 0.88 - with data instances removed.

Table 6.3: Un-tuned models results, trained on swim data, evaluated against swim stroke labels using repeated K fold cross-validation

Model	Average Score	Standard deviation of scores
KNN	0.92	0.02
SVM RBF	0.94	0.02
RFC	0.94	0.02

To further increase the performance, oversampling techniques were considered. SMOTE, as detailed in chapter 2 section 2.2, is used to balance the dataset hence increasing the total number of samples. Applying SMOTE and repeating the cross-validation, generates the results shown by table 6.4. As seen here, employing SMOTE improves model performances and reduces the standard deviation of scores (indicating less likely to over fit).

Table 6.4: Un-tuned models results, trained on oversampled swim data, evaluated against swim stroke labels using repeated K fold cross-validation

Model	Average Score	Standard deviation of scores
KNN	0.95	0.011
SVM RBF	0.96	0.01
RFC	0.97	0.001

Thus far, no specific hyperparameter tuning has been performed, instead the default configurations have been used (SVM uses the RBF kernel by default as per by the Scikit learn python package). It was assumed that performance could likely can be increased via adjusting the appropriate model hyper parameters. KNN only has 1 main hyperparameter: 'number of neighbours'. Hence, cross-validation (identical configuration used as described above) is applied incrementing the 'number of neighbours' hyperparameter and recording the score/standard deviation of scores per increment. The results are demonstrated by table 6.5. As demonstrated 'number of neighbours' configured as 1 improves the score (0.97) compared to the default configuration (5).

Table 6.5: KNN tuning results, trained on oversampled swim data, evaluated against swim stroke labels using repeated K fold cross-validation

Number of neighbours	Average Score	Standard deviation of scores
1	0.97	0.01
2	0.95	0.01
3	0.96	0.01
4	0.95	0.01
5	0.95	0.01

Unlike KNN, SVM and RFC have multiple hyperparameters to investigate hence cross-validated grid search methods were used. This allows automated comparison of multiple hyperparamters and determines which configuration achieves the best result. As such, ta-

ble 6.6 demonstrates the SVM hyperparameters investigated and the configuration of such parameters that achieve the best score (0.97: weighted F1 score , standard deviation of scores: 0.01). Table 6.7 repeats this process however for RFC model detailing the configuration that gives the best score (0.97: weighted F1 score , standard deviation of scores: 0.01). As with proper configuration of KNN, hyperparamter tuning achieves improved performance for the SVM and RFC models. Therefore KNN, SVM and RFC are trained on the oversampled data using the discussed settings.

Table 6.6: SVM best parameter selection determined via cross-validated grid search

Hyper parameter	Value
C	100
Coef0	0
Degree	1
Gamma	Scale

Table 6.7: RFC best parameter selection determined via cross-validated grid search

Hyper parameter	Value
Number of estimators	200
Criterion	Gini

In addition to the classical machine learning methods discussed above, a neural network approach is also employed. The optimum model configuration is demonstrated by figure 6.10. As shown here this model uses dense neurons considering the use of tabular data, as opposed to the sequence data used in chapter 5. The activation function used by all layers in this network is the ELU function. Batch normalisation and dropout (set at 20%) is employed after each dense layer for training/regularisation purposes, as per chapter 2.

As with previous neural network investigations in this thesis, the training set is split into train and validation subsets so that early stopping can be used (patience set as 10). The sparse categorical entropy loss function is used which is optimised via the ADAM optimiser. Table 6.8 details the specific configuration of dense each neuron layer in the model. As demonstrated, the output layer has 4 neurons matching the number of unique swim strokes to be classified in the data set.

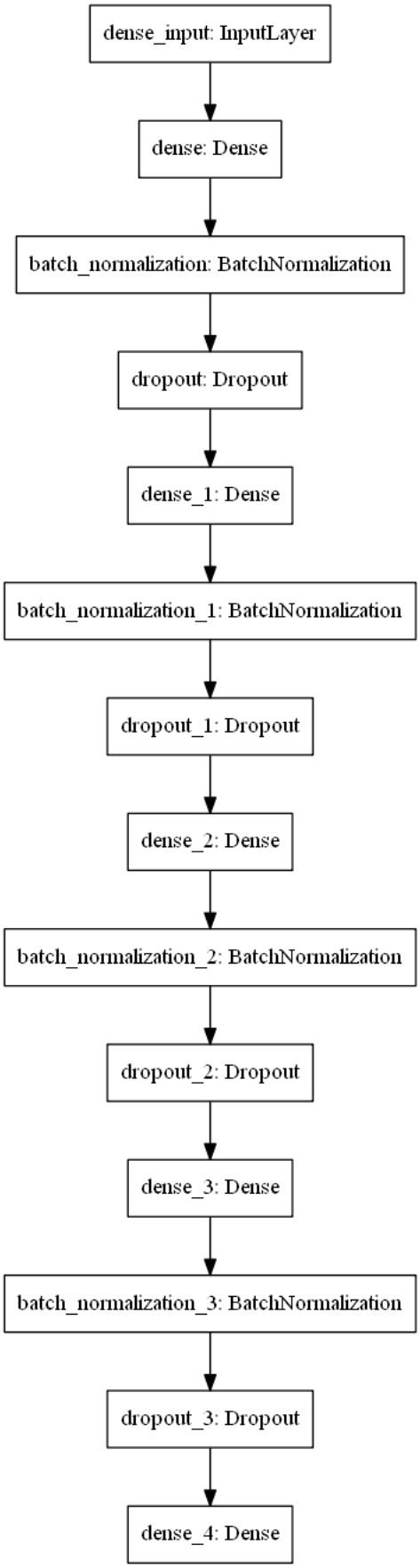


Figure 6.10: Neural network structure used for swim stroke classification.

Table 6.8: Neural network dense layer configuration details.

Layer	Number of neurons	Activation	Kernel constraint (Max norm)
1	30	ELU	3
2	20	ELU	3
3	10	ELU	3
4	5	ELU	3
5 (output)	4	Softmax	3

K fold cross validation is not performed using this neural network structure, as was done above with the classic machine learning models. Instead, an ensemble method (similar to the approach used in chapter 5), is used and will be detailed below.

Model testing

All training, tuning and validation discussed above is performed on swim accelerometer data sampled from one subject. To investigate how such machine learning models performed on other subjects, samples from 4 participants were tested. Tables 6.9, 6.10 and 6.11 demonstrate the results of the previously trained models - KNN/SVM/RFC respectively - applied to this testing data.

Table 6.9: KNN performance tested on swim data from 4 subjects.

	Precision	Recall	F1-score	Count
Breast stroke	0.95	0.97	0.96	239
Front crawl	0.94	0.97	0.96	188
Back stroke	0.94	0.97	0.96	101
Not swimming	0.54	0.27	0.36	26
Macro average	0.84	0.79	0.81	554
Weighted average	0.93	0.94	0.93	554

Table 6.10: SVM performance tested on swim data from 4 subjects.

	Precision	Recall	F1-score	Count
Breast stroke	0.95	0.95	0.95	138
Front crawl	0.93	0.74	0.83	102
Back stroke	0.97	0.91	0.94	101
Not swimming	0.24	0.65	0.35	26
Macro average	0.77	0.81	0.77	554
Weighted average	0.91	0.86	0.88	554

Table 6.11: RFC performance tested on swim data from 4 subjects.

	Precision	Recall	F1-score	Count
Breast stroke	0.97	0.98	0.98	239
Front crawl	0.98	0.96	0.97	188
Back stroke	0.97	0.98	0.98	101
Not swimming	0.82	0.88	0.85	26
Macro average	0.94	0.95	0.94	554
Weighted average	0.97	0.97	0.97	554

As demonstrated by these results, the RFC model better classifies the swim data rel-

ative to swim stroke. The performance of the KNN and SVM models are inhibited by poor 'not swimming' classification ability which is a detriment to the overall performance. Having performed this analysis, the neural network model tested. As mentioned above, an ensemble classifier is constructed. As before, multiple neural network models (matching the configuration demonstrated by figure 6.10 and table 6.8) are trained on different train validation splits (30 splits in total) of the training data. These 30 models form the ensemble that generates an aggregated set of predictions for the test set. The aggregation is such that the class probabilities, outputted by each individual model per test instance, are summed. The class associated with the highest summed probability is classified for the given sample. The classification report for this ensemble classifier is demonstrated by table 6.12.

Table 6.12: Ensemble neural network performance tested on swim data from 4 subjects

	Precision	Recall	F1-score	Count
Breast stroke	0.98	0.98	0.98	239
Front crawl	0.97	0.97	0.97	188
Back stroke	0.97	0.97	0.97	101
Not swimming	0.88	0.85	0.86	26
Macro average	0.95	0.94	0.95	554
Weighted average	0.97	0.97	0.97	554

The performance of the neural network ensemble classifier is similar to the performance noted from the RFC classifier. This is reasonable considering a RFC classifier is an ensemble method itself. Chapter 1, section 1.3.6 discusses existing underwater stroke classification algorithms the best of which achieves 99% accuracy for classifying breast stroke, front and

back crawl [87]. F1 score is not detailed hence direct comparison to the work demonstrated in this chapter is difficult. However, undeniably, the performance demonstrated by Zhang et al appears good. The issue being that six IMUs are required compared to the single accelerometer used by the approach demonstrated in this chapter.

6.2.5 Breathing/Stroke Rate Analysis

Reliable swim stroke classification is a useful tool as it can be used to assist the estimation of swimming breathing/stroke rate from the accelerometer data. If the the swim stroke is known, analysis of the appropriate axis can be used to estimate breathing rate. For example, during breast stroke, the sinusoidal rhythm of the z axis likely correlates to breathing rate considering the mechanics of the stroke. Likewise, during front crawl, breathing rate can be estimated from the sinusoidal rhythm of the x axis considering breaths are taken periodically as the head rotates out of the water. Estimation of front crawl breathing rate via this approach is found to be viable independent of how many strokes are taken before the swimmer takes a breath. This is because the body rolling out of the water so that the swimmer can breathe, generates the most significant movement (see figure 6.21), and this is represented in the frequency domain. Back stroke was theorised to be more difficult considering a swimmer can breathe whenever considering the mouth does not go underwater, and therefore may not correlate to a particular body movement. However, as discussed in section 6.2.1 its possible that swimmers may sub consciously synchronise breathing with the rotation of the arms and hence breathing rate will correspond to x axis rotation. To generate a breathing rate estimate from the appropriate accelerometer channel, the same process used to generate breathing rate from the breathing rate sensor is used (peak value of the FFT processed data). To investigate this, table 6.13 demonstrates the average difference between the number of breaths estimated (per 5 second sample),

from the appropriate accelerometer channel, compared to the number of breaths estimated (per 5 second sample) from from the dedicated breathing rate sensor.

Table 6.13: Average differnece between breathing estimated from acclerometer axis vs dedicated breathing sensor.

Swim stroke	Average difference
Breast stroke	0.72
Front crawl	0.58
Back stroke	0.68

As demonstrated, on average the difference in estimate using each approach (accelerometer vs breathing sensor) is less than one breath. However, this error is for a 5 second window hence is a significant factor. For reliable estimation a longer window is recommended, however the demonstrated configuration may be suitable for identifying magnitude changes as previously discussed in this thesis.

Similar to this principle, by observing the appropriate channel, stroke rate can also be determined. This process also involves applying FFT to the appropriate accelerometer channel (x axis if front crawl/back stroke, z axis if breast stroke) to identify the dominant sinusoidal component. The frequency of this component, similar to breathing, identifies the stroke rate.

6.2.6 Combined Analysis

To demonstrate an example of how the techniques devised in this chapter and chapter 5 can be used in unison; figure 6.11 shows the underwater BPM generated by the CNN-RNN model (from chapter 5) overlayed with the swim stroke classifications recorded during a period of swimming. Specifically, for this example, the RFC model (see section 6.2.4)

is used and as demonstrated sensibly classifies swim stroke relative to changes in heart rate. Initially, there is an increase in heart rate during front crawl swimming followed by a reduction during breast stroke. This is expected considering front crawl generally involves more energetic movements, logically BPM drops during the period of not swimming.

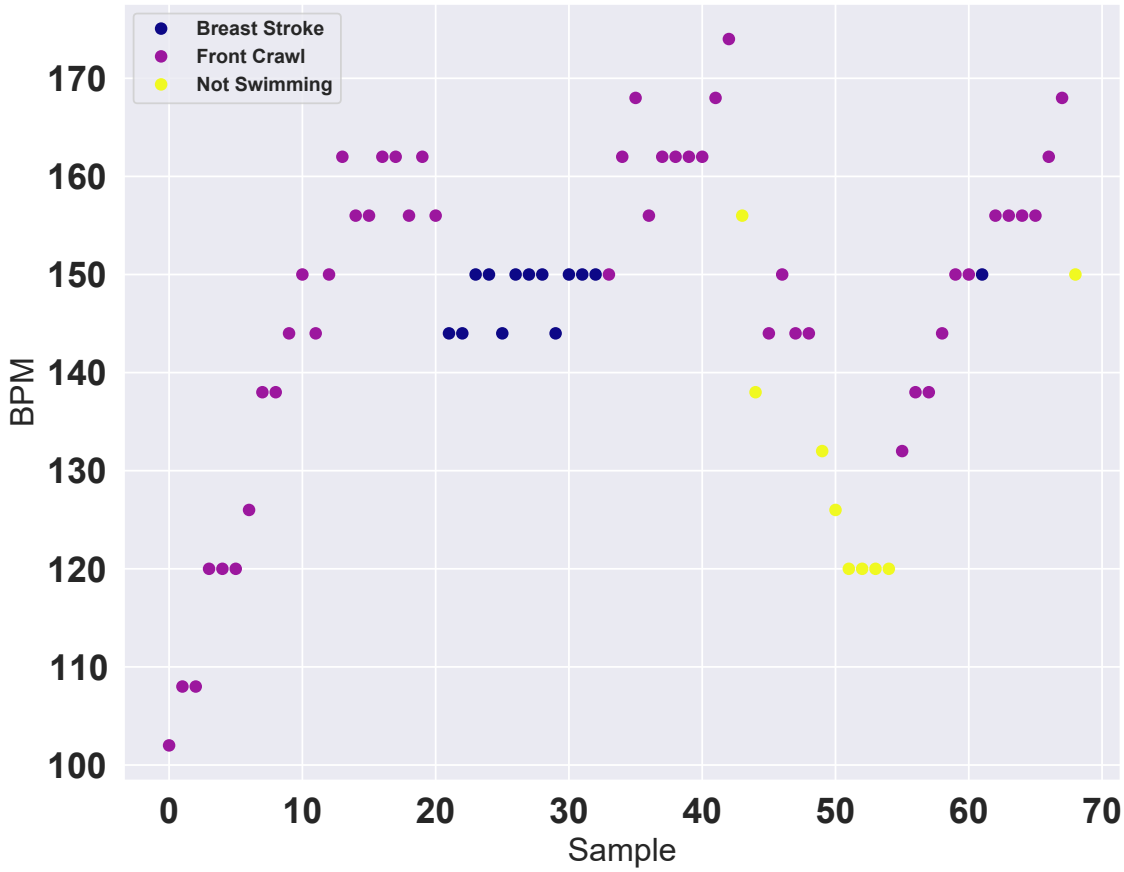


Figure 6.11: Swimming BPM (generate via methods detailed in chapter 5), colour coded by swim stroke.

Figure 6.11 shows an example of how the information generated could be presented

to, for example, a swim coach. Considering all discussed techniques thus far have been developed from data sampled in a pool, this style of presentation was an obvious choice. Application to open water swimming would would also be suitable for this form of analysis however some modifications may be necessary. Particularly if used in the sea where the motion caused by waves may need to be accounted for due to the effect this would have on the accelerometer readings. This would required additional training data that featured the motion artefacts caused by waves so to investigate whether the swim strokes could still be distinguished. Considering the relatively small training set used in this chapter to generate trained models, this may not be a overly time consuming process.

6.3 Conclusions

This chapter investigates the analysis of the data recorded by the underwater sensor system and the results generated by assoicated machine learning technqiues. Primarily, the viability of four main machine learning algorithms for classifying swim stroke, using features generated from sampled chest accelerometer data. As demonstrated, the RFC and ensemble classifier in particular can generalise well as shown by the classification testing demonstrated above. The training set, is relatively small in size however effective models can still be trained from this set. Increasing the size of the training set could possibly further improve the test performance which is plausible considering the performance increases that was noted when SMOTE was used. Swim stroke classification is then used to assist the estimation of breathing/stroke rate solely from the chest accelerometer data. The results of this estimation are compared to a dedicated breathing rate senor which appear to demonstrate agreement. Use of the swim classification techniques, is then demonstrated in unison with BPM generation techniques introduced in chapter 5 to provide analysis of swimming. Applications to open water are then discussed and potential deplyoment cases

are highlighted.

Chapter 7

Sleep Stage Monitoring for Microgravity

7.0.1 Introduction

Chapter 1, sections 1.3.5 and 1.2.1 introduces both sleep and sleep in space. Crucially, some of the difficulties associated with sleep in microgravity are discussed and how poor sleep quality could be dangerous for astronauts operating in microgravity. In addition the principles of PSG are discussed, which is the gold standard for sleep analysis a crucial component of which is sleep stage analysis. PSG utilises a large array of sensors, some intrusive and potentially uncomfortable. These factors result in a full PSG setup being unsuitable for convenient deployment into microgravity considering the weight calculations associated with space flight and the already uncomfortable nature of sleep in space. Furthermore, it is also important to ensure specific components i.e heart rate sensor/accelerometers have good contact with the subject considering the weightless conditions. Here, the sensitivity of the sensors is important so that good signal interpretation can be achieved. Generally, the analysis performed here allows assessment of the sensor platforms response to small

signal variations considering the minor physiological variations assoicated with each sleep stage.

This chapter discusses the results obtained from the constructed sleep sensor variant and associated sleep stage prediction algorithm which is developed based on the findings of this chapter. The device is intended to be a sleep diagnostic tool to be used by researchers in the field of space sleep study. As highlighted in chapter 3, the sensor system utilises multiple sensors. This approach allows for dimensionality reduction techniques to be applied to the sampled data and good cluster separation to be achieved, allowing analysis which in turn lead to the development of the 'automatic algorithm'. The sensor platform has been designed considering potential deployment into microgravity. Primarily the sensors are mounted onto an elasticated chest strap so to maintain contact with the subject. As highlighted previously good sensor contact is required for wearable sensor study in space considering the microgravity environment. Also, the sensor system is designed to be as lightweight as possible. This is done in attempt maximise the comfort associated with wearing the system considering the discussed uncomfortable experience of sleeping in space. Regarding accelerometer calibration, no on chip steps are taken with regards to negating acceleration due to gravity, a factor obviously not present in space. However during data processing, regardless of testing on earth or any potential future deployment into space, the accelerometer signal baseline (acceleration due to gravity in this case) is removed via software. This signal baseline would not be present in space hence in this case the baseline removal would have no effect on the signal. In addition, the use of WiFi for data transfer warrants future investigation into telemedicine applications where internet communication, using WiFi technology, enables remote monitoring capabilities. This may be particularly useful in future spaceflight operations where humans spend increased time in space, potentially in isolated circumstances. Key findings of this chapter are the good

performance achieved by the devised sleep stage prediction methods when applied to data sampled from one participant.

7.1 Methods

7.1.1 Data acquisition process

The sensor system variant used to generate the results discussed in this chapter is detailed by chapter 3, seciotn 3.2.2. As discussed the device focuses on ECG, breathing, changes in skin temperature and accelerometer data.

To allow initial verification of the sleep analysis approach, 33 data sets (33 nights of sleep) have been acquired and used in the analysis presented here. All data presented has been obtained from a single user, in accordance with relevant ethical approval processes in place at the time of testing. As per chapter 3, data is recorded by the sensor system and transmitted every 10 seconds. For the results demonstrated in this chapter, data is then averaged (via mean) over a minute. All data was normalised on per-record basis via a z-score normalisation to ensure a consistent range was present in all cases. An example normalised dataset is shown by figure 7.1.

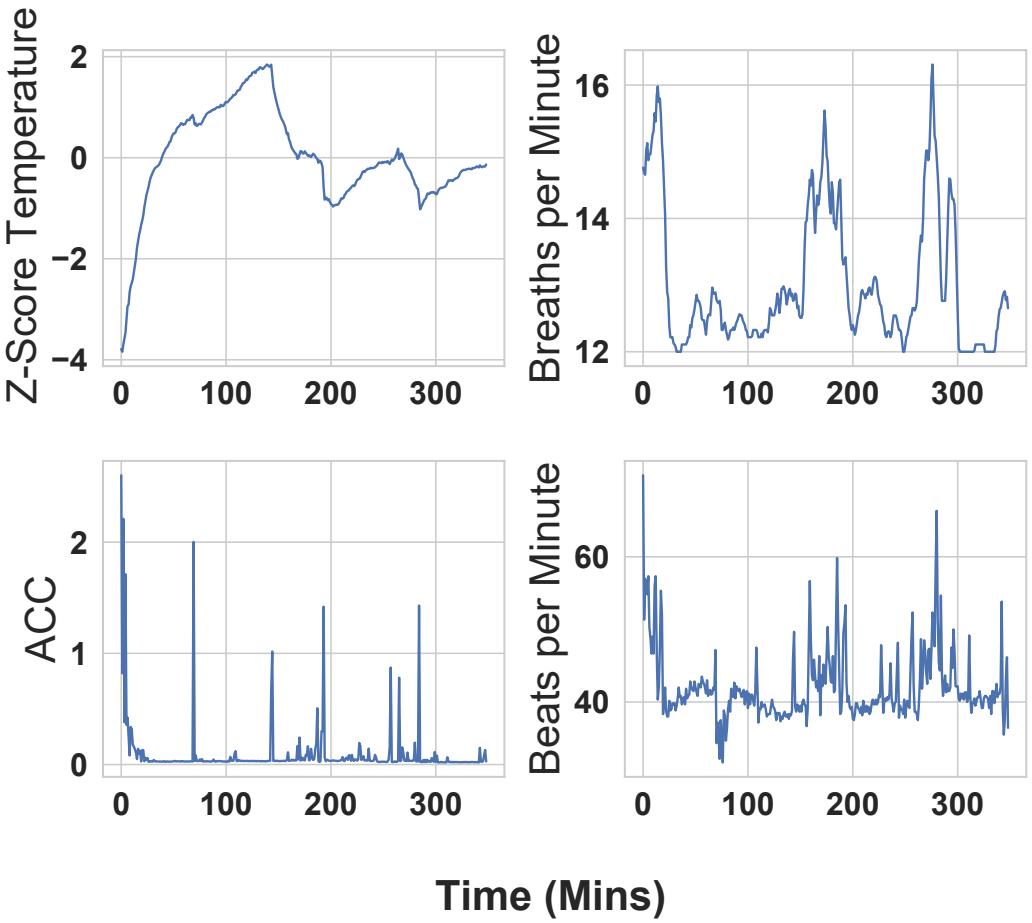


Figure 7.1: Example normalised (z-score) data recorded by sleep sensor system.

Naturally, there are physiological differences between different users, however this chapter focuses on studying the intra-user variability to show that robust operation is possible as a proof of principle. Participant-specific staging algorithms could be developed if necessary in future works if the automated classification method is affected by inter-user variability. To benchmark the results, and to show comparable performance to consumer grade wearables, the sleep stage predictions made by the discussed algorithm are compared to the stage classifications from a Fitbit Versa 2 smartwatch. As discussed in chapter 1, this

smartwatch utilises two sensors: a 3-axis accelerometer and an optical heart rate monitor based on PPG [164]. Heart Rate Variability (HRV) is calculated from the heart rate data. Fitbit state that sleep stage tracking is performed via analysis of acceleration, heart rate Beats Per Minute (BPM) and HRV data [164]. The sleep stage estimations are made post sleep when the smartwatch is synced to smartphone via Bluetooth [63, 164]. As such, this chapter assesses the performance of the constructed sensor platform and associated sleep classification algorithm compared to the commercially available Fitbit device. Whilst this device has been benchmarked against PSG, it is not the gold standard approach. Hence, this chapter is intended to provide an overview for the capabilities of a low cost sensor platform (and sleep classification algorithm), against a readily available, consumer grade sleep tracker and how the introduction of different sensing capabilities may affect classification performance. Chapter 8 considers the investigation of comparing the sensor system/devised algorithms against the gold standard PSG.

7.1.2 Data analysis

Sleep stage analysis is performed offline, post sleep, in line with other commercially available systems. It is envisaged that adaptation to real time sleep stage analysis is easily possible in future work. The initial principle of sleep stage identification method was to perform dimensionality reduction on the data. It was found that only four of the attributes were necessary for this approach: breathing rate; heart rate; temperature and temperature gradient. Raw acceleration was excluded from the clustering analysis due to its relatively infrequent variations.

Dimensionality reduction was initially performed via PCA (see chapter 2, section 2.3.1). The clusters apparent from application of PCA were identified via KMeans (see chapter 2, section 2.3.1) where K was set to four in order to match the number of sleep stages to be

identified (4). Analysis was performed by comparing clusters located via KMeans to the sleep stage predictions from a Fitbit smartwatch. Timestamps were used to identify which Fitbit stage predictions corresponded with the data points recorded by the constructed sensor platform. This approach allows comparison between the KMeans clusters and the Fitbit predictions projected onto the 3D PCA plots. The same colours have been used to show each stage regardless of assignment method for ease of understanding. This colour coding was then applied to the corresponding processed physiological data recorded by the sensor system. Which allowed the assessment and understanding of significant signal variations that corresponded to different sleep stages. This comparison, when performed over a large number of data sets, has allowed a Python program to be devised that automatically identifies the sleep stage without the need for data dimensionality reduction. The program is a series of conditional statements that interprets the processed sensor data collected by the system. This program is based upon evidence in the literature on sleep stage physiology and the observations made from the clustering/dimensionality reduction, allowing the sleep stage to be determined [14, 34].

7.2 Results and discussion

7.2.1 Cluster identification from dimensionality reduction

As stated previously, use of dimensionality reduction allows visualisation in alternate coordinate spaces and can increase data separation. Figure 7.4 details an example of how data was initially examined by reduction into a lower dimensionality coordinate space. This example is a representation in 2D of the data demosntrated by figure 7.1. Shown here is a 3D coordinate space displayed as individual 2D projections (i.e. x-y, x-z), so that the cluster separation can be clearly seen. The axes for these plots represent the principal components

produced by PCA. KMeans has been applied to this scatter plot and has appropriately identified the clusters within the data.

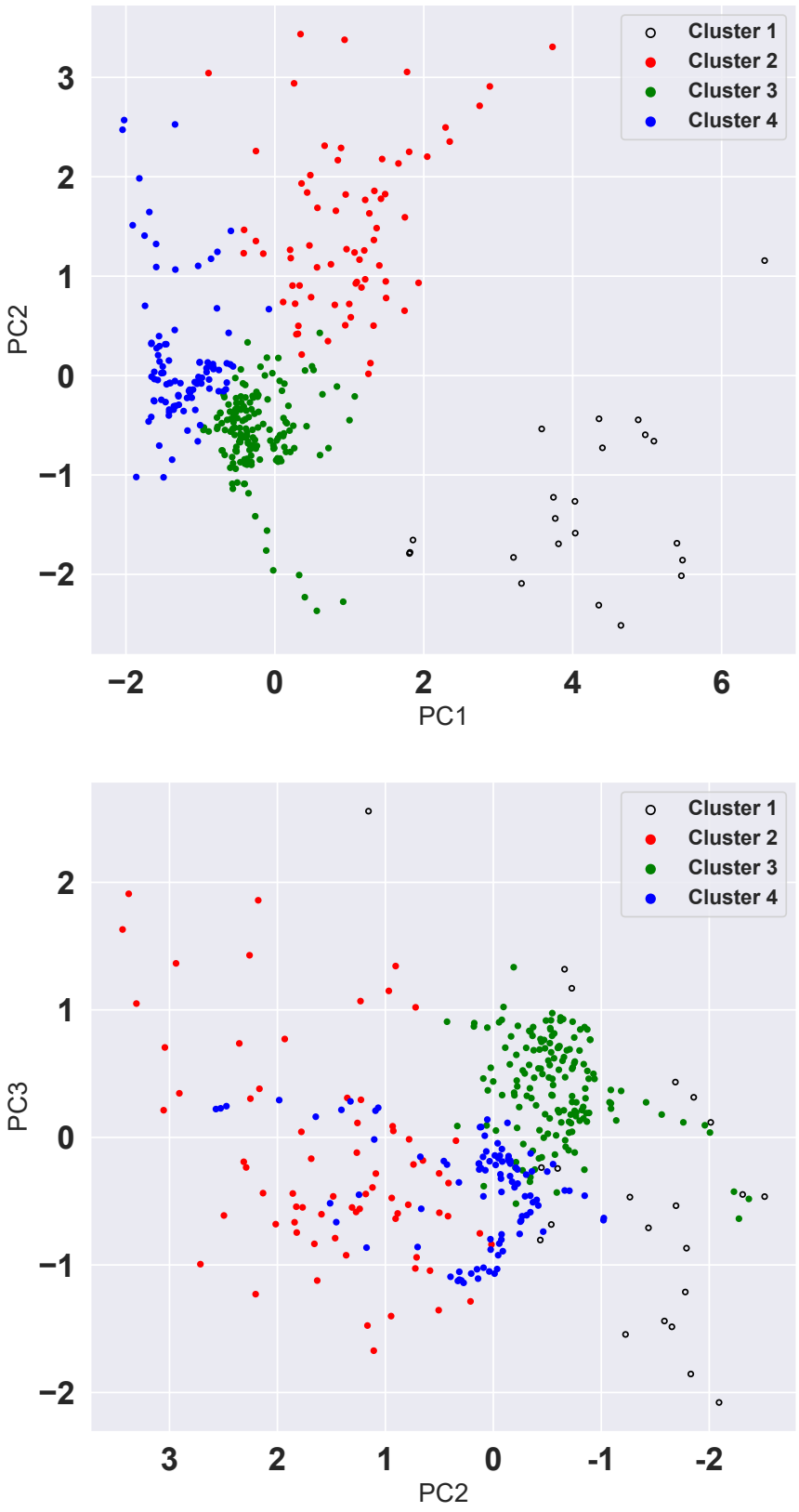


Figure 7.2: Data recorded by sleep system, coloured coded via KMeans clustering.

Figure 7.3 demonstrates an example of sleep stage predictions made by a Fitbit smartwatch (recorded during same night as the data demonstrated by 7.1) over a period of approximately 350 minutes of sleep. This graph has produced by downloading the Fitbit sleep stage predictions and associated timestamps so that a representation of the sleep structure for this night can be plotted. To understand if the KMeans clusters demonstrated by figure 7.2 correspond to individual sleep stages, figure 7.4 shows the same principal components (same as figure 7.2), now colour coded by the Fitbit labels. It is important to note, although the Fitbit predictions have been overlayed onto this data which was sampled from the chest strap and processed via PCA, the Fitbit predictions are generated based on wrist (watch) data. However, this process was important as it allowed correlation between the clusters produced by KMeans to the sleep stage predictions made by the smartwatch. Specifically, from figure 7.4 it is seen that the concentrations of deep and light sleep points are split between two clusters, located in the central region. KMeans however delineates these into two separate clusters, as shown by figure 7.2. While some individual samples are different, the broad structure of the sleep stages is clearly being extracted by the KMeans clustering applied to the chest data.

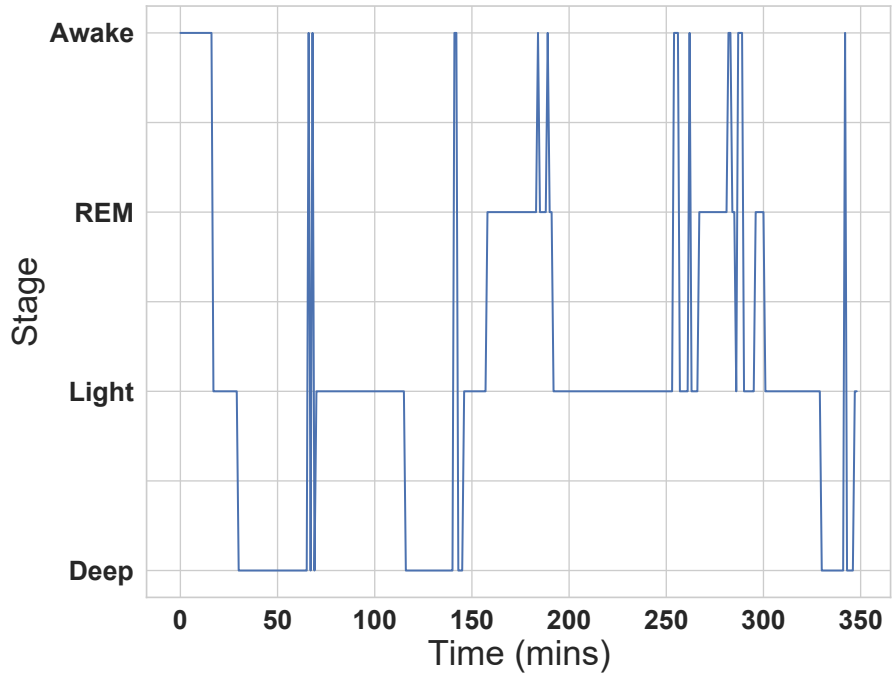


Figure 7.3: Example of the sleep stage predictions made by a Fitbit smartwatch.

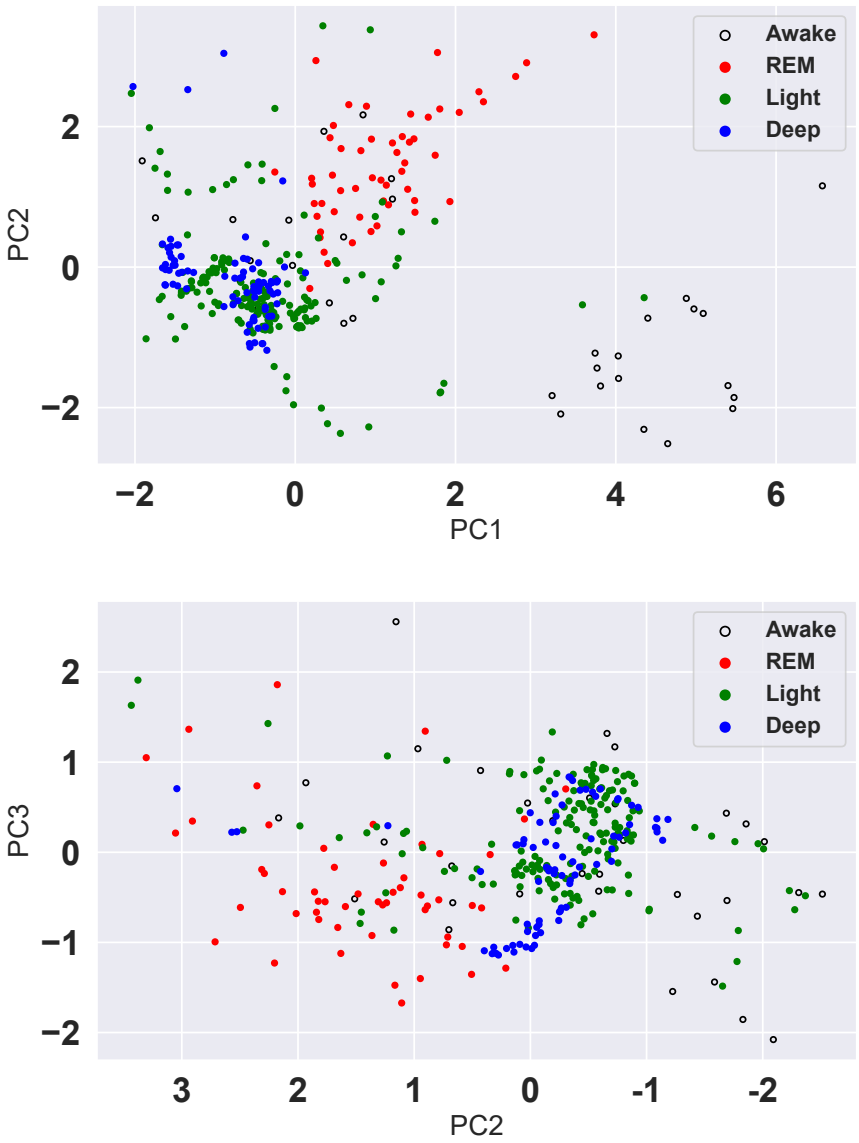


Figure 7.4: Processed physiological data recorded by sleep system, coloured coded via the Fitbit predictions.

Figure 7.5 shows this process in the time domain. Here the output from the sleep system sensors (not Fitbit sensors) have been overlayed with the predictions from the Fitbit smartwatch. This highlights how the physiological signals measured at the chest

correspond to the different stages identified by the Fitbit. As demonstrated, in this case, the variations in the sensor data relate to the appropriate physiological changes associated with the sleep stages (eg increased BPM and BR for REM).

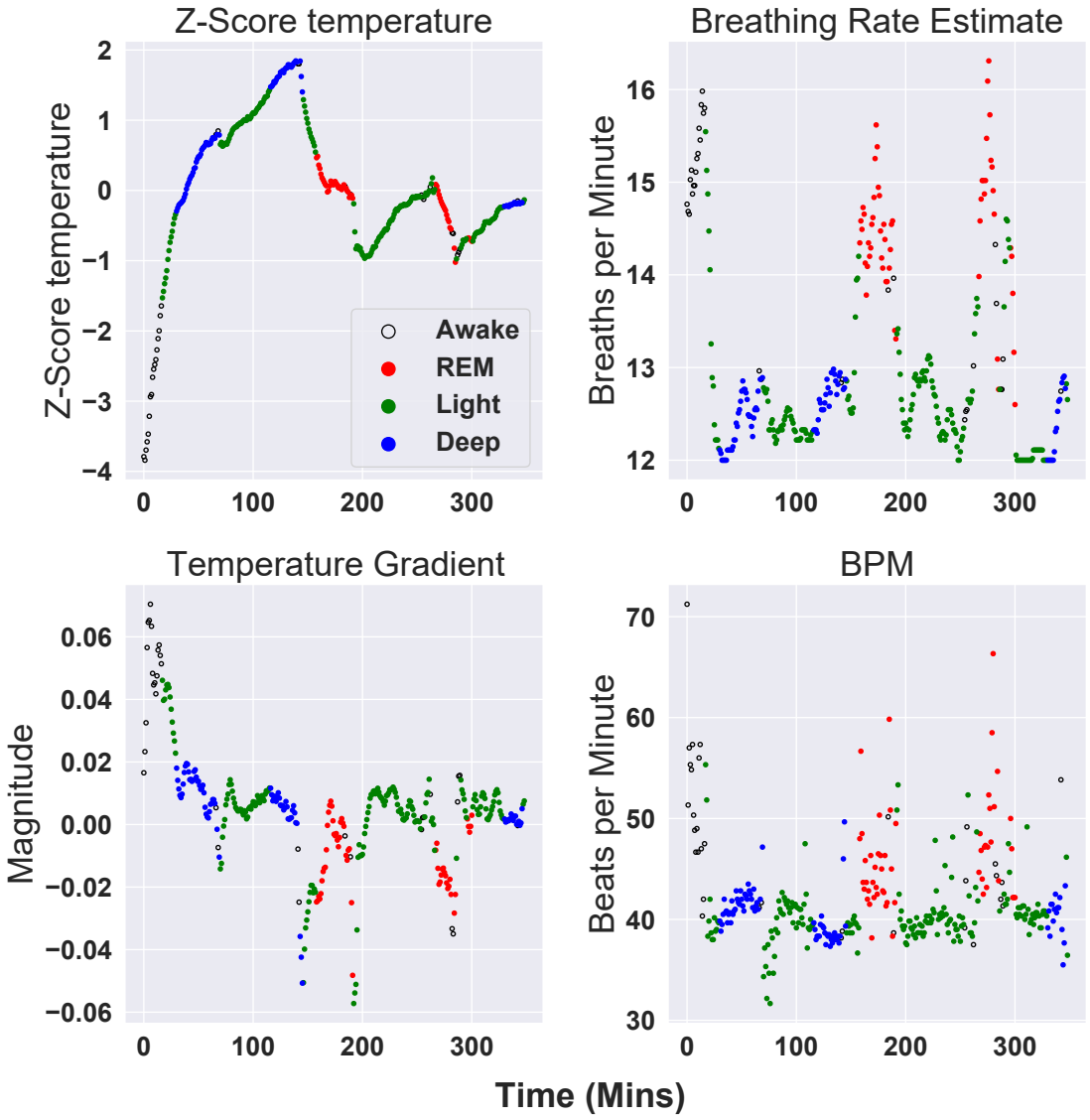
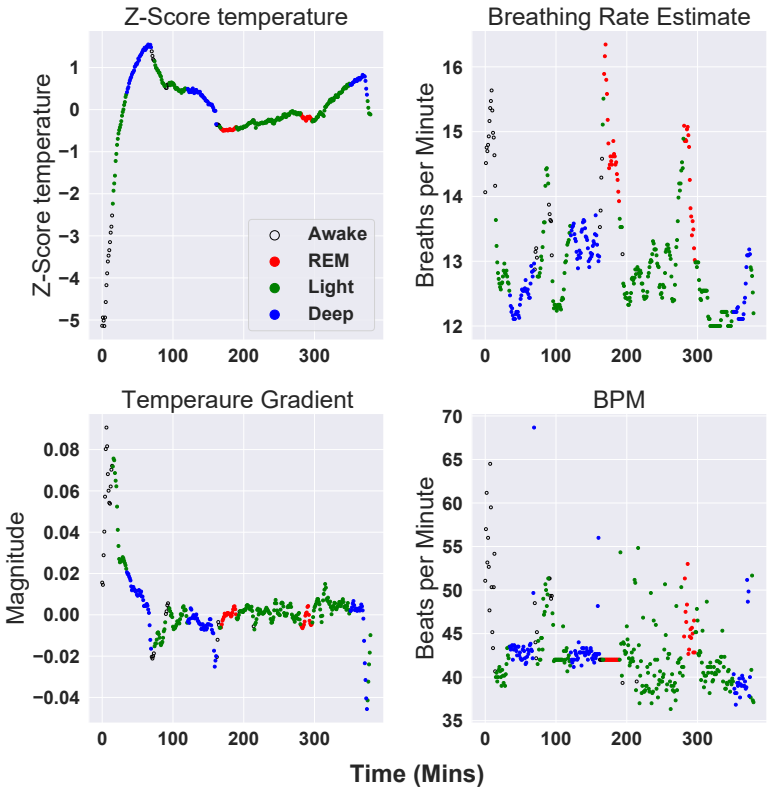


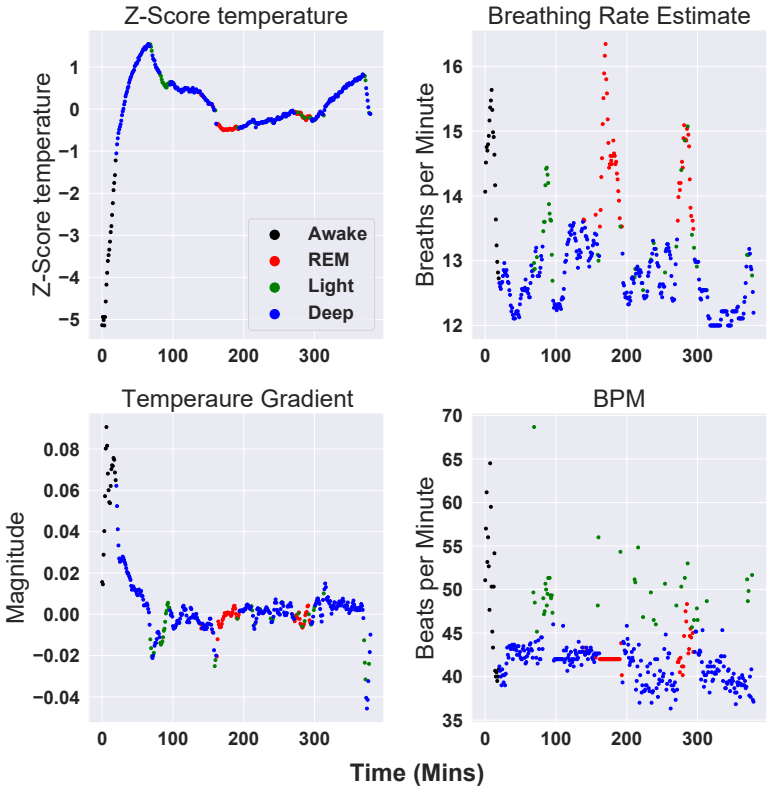
Figure 7.5: Data recorded by sleep system in time domain, colour coded via Fitbit predictions

For the example sleep data demonstrated by figures 7.2/7.4. The disagreement observed may indicate regions where the Fitbit predicts an incorrect sleep stage(s). Analysis of multiple sleep datasets would allow confirmation of this theory. However in some cases it was apparent, considering the known structure of sleep as per literature, that the KMeans process highlighted clusters that did not accurately indicate the sleep stages. Figure 7.6 demonstrates the results of clustering (k=4, same strategy as previously employed) a PCA processed dataset collected on a separate night (compared to figures 7.2/7.4/7.5) of sleep monitoring. Here poor correlation between Fitbit and KMeans is demonstrated, in this case the KMeans clustering results (figure 7.62) cannot be reasonably compared to the predictions made by the Fitbit (figure 7.61) given the significant disparity. Here and with other instances where the KMeans predictions are thought to be wrong, compared to both the Fitbit and sleep physiology, using LDA (see chapter 2 section 2.3.2) can be beneficial. Figure 7.7 demonstrates the sensor data (same physiological data used by figure 7.6) colour coded according to the generated predictions from application of KMeans to the LDA (applied using Fitbit labels) results. As demonstrated, there is significant improvement with regards to the correlation between the two methods. However, the use of LDA (and any other supervised technique) requires the use of labels associated with the data. The only labels available in this case are the Fitbit labels which have associated errors affecting the performance of the supervised technique. However, demonstrated in the overall results section of this chapter, despite potential erroneous labels generated by the Fitbit device, it is shown that the use of LDA-KMeans compared to PCA-KMeans yields better correlation to the devised automatic method. Regardless, the use of labels is not ideal considering the use of Fitibt predictions to 'influence' clusters. However, the clustering process (KMeans) remains unsupervised irrespective of dimensioanlity reduction approach. Ultimately, studying a combination of both PCA and LDA reduced datasets led

to the design of the automated approach.

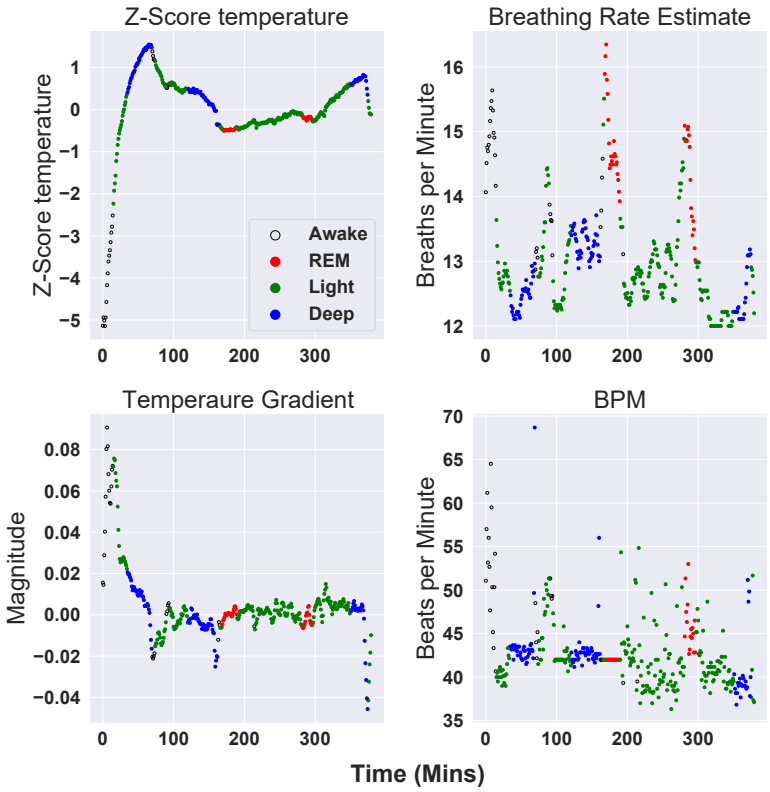


(1)

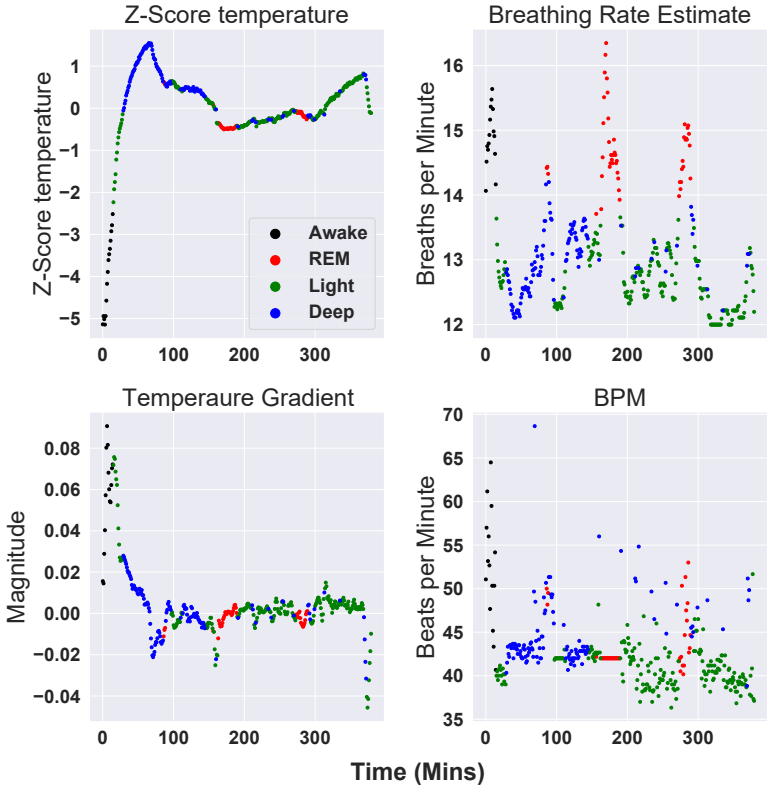


(2)

Figure 7.6: Sleep data (different data compared to as figures 7.2/7.4/7.5) shown in the time domain coloured by 7.61 Fitbit and 7.62 PCA-KMeans. As demonstrated, there is an apparent over prediction of deep sleep for the KMeans approach.



(1)



(2)

Figure 7.7: Sleep data (same as figure 7.6) shown in the time domain coloured by 7.71 Fitbit and 7.72 LDA-KMeans. Improved correlation between these methods compared to PCA (figure 7.6).

7.2.2 Automated approach

Application of KMeans (as shown above) demonstrated that individual sleep stages can be identified from the chest recorded data. However the process is not fully automated as KMeans does not associate a sleep stage to each cluster identified. The sleep stage can be identified by comparison to the predictions provided by the Fitbit device, as discussed above. To remove the need for the Fitbit device, a logical prediction function was made in Python that identifies the trends seen in the processed data. As stated above, this was termed the Automatic method and it makes use of all of the available data streams. Specifically, this program works by using *If* statements that assess whether multiple parameters have met certain thresholds matching rules that a human interpreter would use. In attempt to ensure this approach is robust, the thresholds are determined by percentile values. This is done so that predictions are not affected by different ranges of data; in particular skin temperature data for which there is high variation in range between recordings. A percentile approach also ensures that there is an element of flexibility to this approach when a different subject wears the device.

Figure 7.8 shows a flowchart detailing the logic behind the automated method. These conditions have been devised based on analysis over 7 nights of data, comparing clusters to sleep literature. An example of this automated process is demonstrated in 7.10 (applied to same data used by figure 7.2).

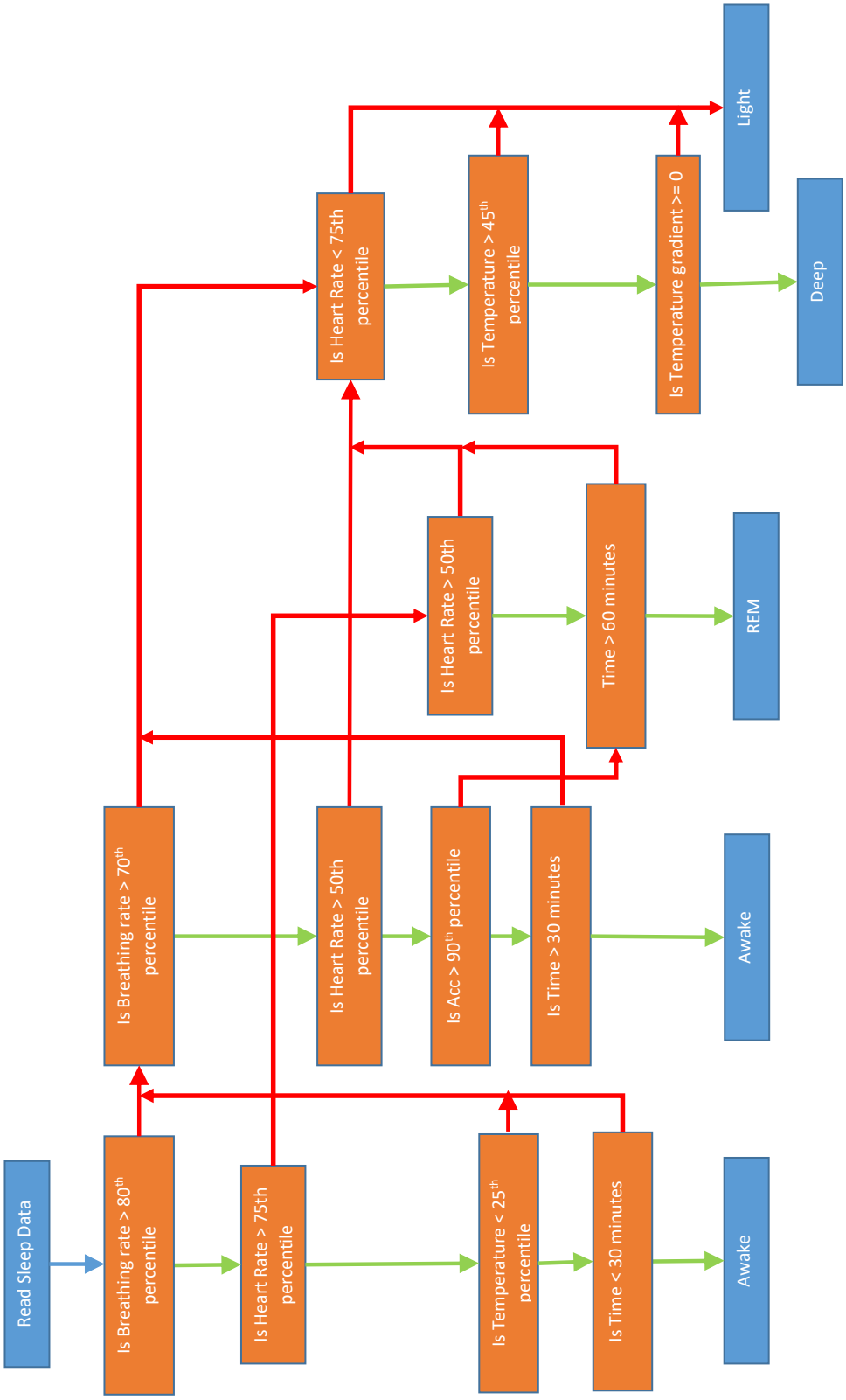
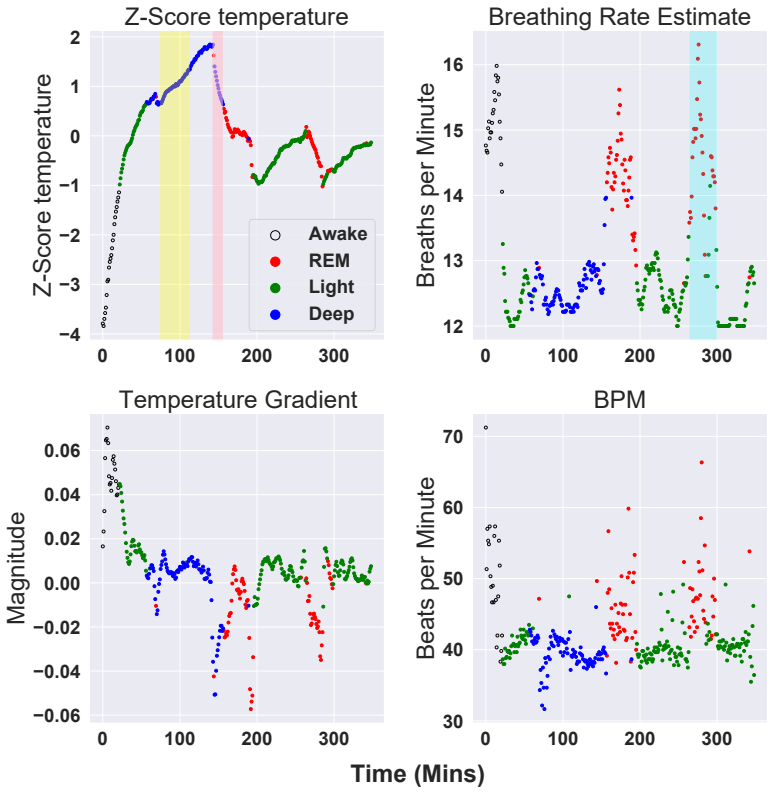
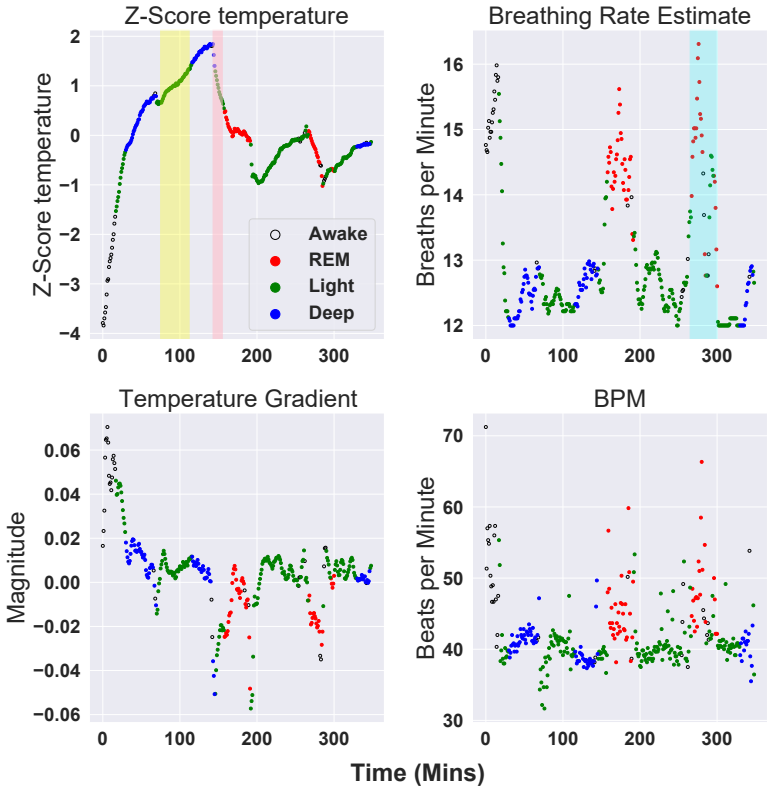


Figure 7.8: Data recorded by our system, coloured by the automatic prediction method.

To demonstrate the benefits of this algorithm figures 7.9/7.10 provide comparison between all sleep stage prediction approaches discussed thus far: KMeans; Fitbit; automatic method. As seen, these graphs have shaded regions. Firstly, the yellow region highlighted indicates a period of deep sleep. As shown, the KMeans approach (figure 7.91) and automatic approach (figure 7.10) are in agreement, however the Fitbit (figure 7.92) is not. This is thought to be because the Fitbit does not monitor temperature data and hence gives an erroneous prediction here. The assumption here is based on the know correlations between NREM sleep and skin/core-body temperature [39].



(1)



(2)

Figure 7.9: Data recorded by sleep system (same night as figures 7.2/7.4/7.5), coloured 7.91 KMeans and 7.92 Fitbit predictions. Regions shaded for discussion in the text.

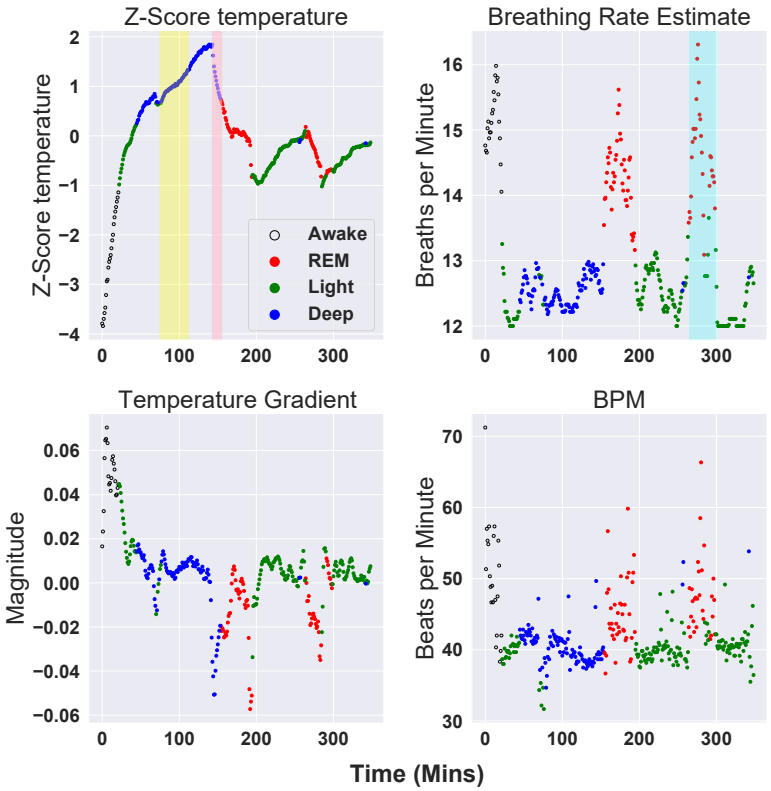
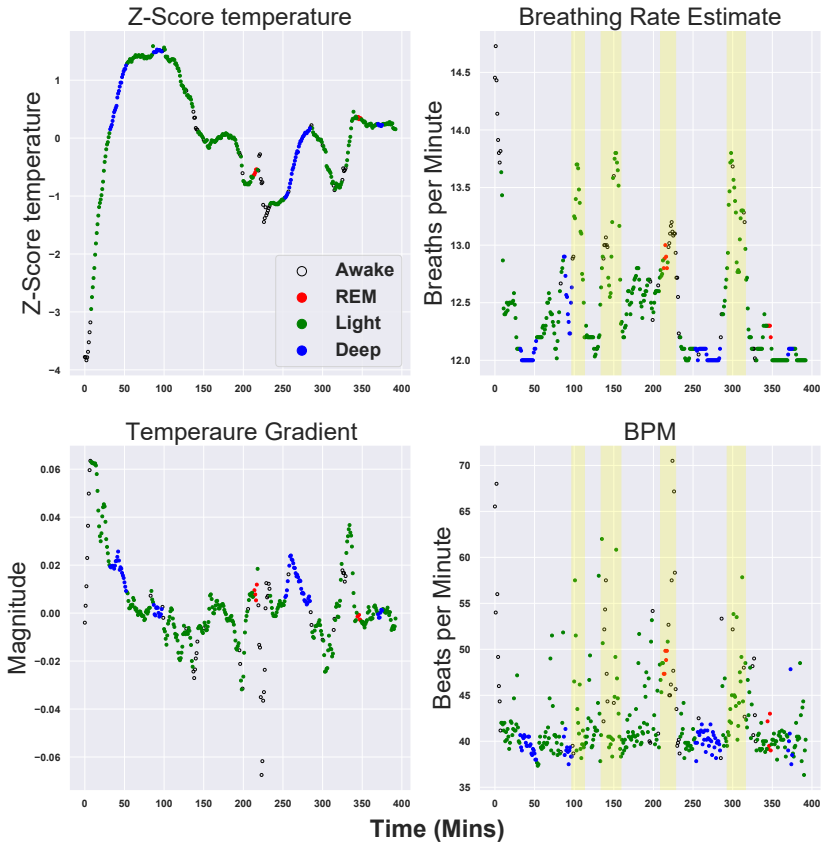


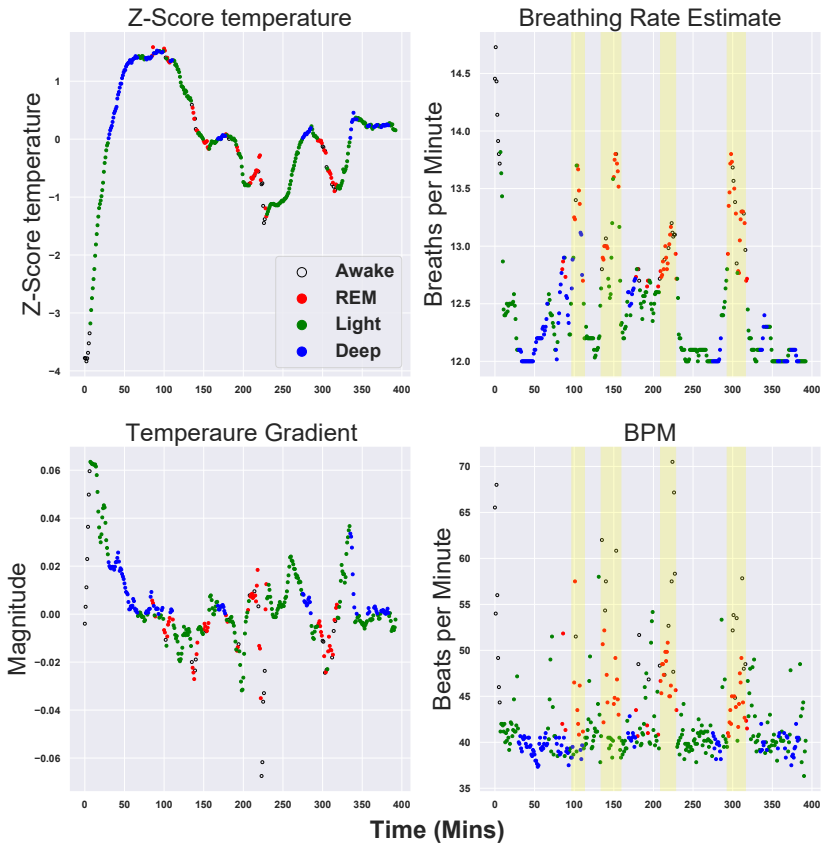
Figure 7.10: Data recorded by sleep system (same night as figures 7.2/7.4/7.5), coloured by the automatic prediction method. Regions shaded for discussion in the text.

The cyan region demonstrates a period of REM sleep. The Fitbit (figure 7.91) has predicted instances of light sleep in this area. Again this is believed to be incorrect and that by including an estimate of breathing rate (not featured on the Fitbit device) this error can be rectified as shown by the automatic method (figure 7.10). Lastly, the pink region demonstrates a correction made by the automatic method (7.10) compared to KMeans (figure 7.91) and Fitbit (figure 7.92). KMeans and Fitbit both predict deep sleep here, however the automatic method predicts light sleep. The automatic method does this as skin temperature is reducing [39]. These factors demonstrate the benefit of using a chest strap worn device that allows inclusion of the extra parameters demonstrated.

To further demonstrate the effectiveness of the constructed sensor platform and associated sleep stage algorithm, figures 7.11/7.12 show other examples of recorded sleep data. The comparison demonstrated by these figures focuses on Fitbit versus Automated sleep stage predictions which are overlayed onto the recorded data. This is done to highlight how the limitations of the Fitbit device can yield incorrect stage predictions. The yellow shaded regions on figure 7.11 demonstrate where the Fitbit predicts light sleep (figure 7.111) compared to the REM predictions made by the automated algorithm (figure 7.112). Within the yellow regions both increased breathing and heart rate can be seen which is known to be associated with REM sleep. The REM sleep predictions made by the automated algorithm are therefore assumed to be valid given the clearly visible, correlated increases in both breathing and heart rate during these periods. As discussed previously, the algorithm used by the Fitbit device to predict sleep stage is not clear. However, it is known that the wrist worn Fitbit does not monitor breathing rate and measures BPM via a PPG sensor both of which could be factors with regards to this issue. Furthermore, figure 7.12 demonstrates a disagreement between the two methods with regards to light vs deep sleep prediction. As demonstrated the Fitbit (figure 7.12) predicts light sleep whereas the the automated method interpreting the sensor data predicts deep sleep. According to the signals demonstrated by the constructed sensor platform, deep sleep seems the correct prediction considering the low BR and BPM correlated to the increased skin temperature. The disagreement between the two approaches is relatively minimal compared to the disparity seen on figure 7.11. However, the under-prediction of deep sleep is a known limitation of the Fitbit from literature and is demonstrated by the overall results section of this chapter.

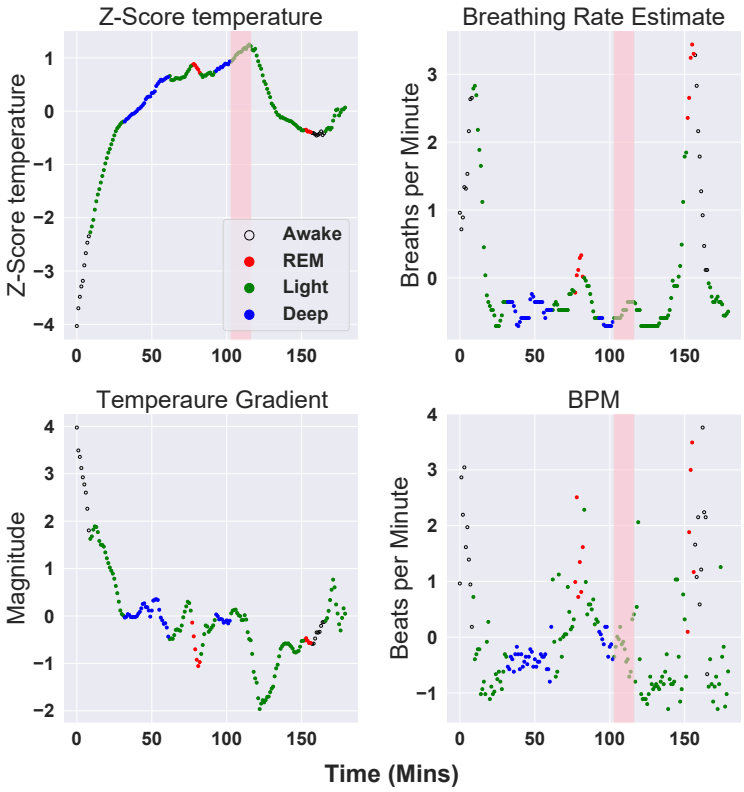


(1)

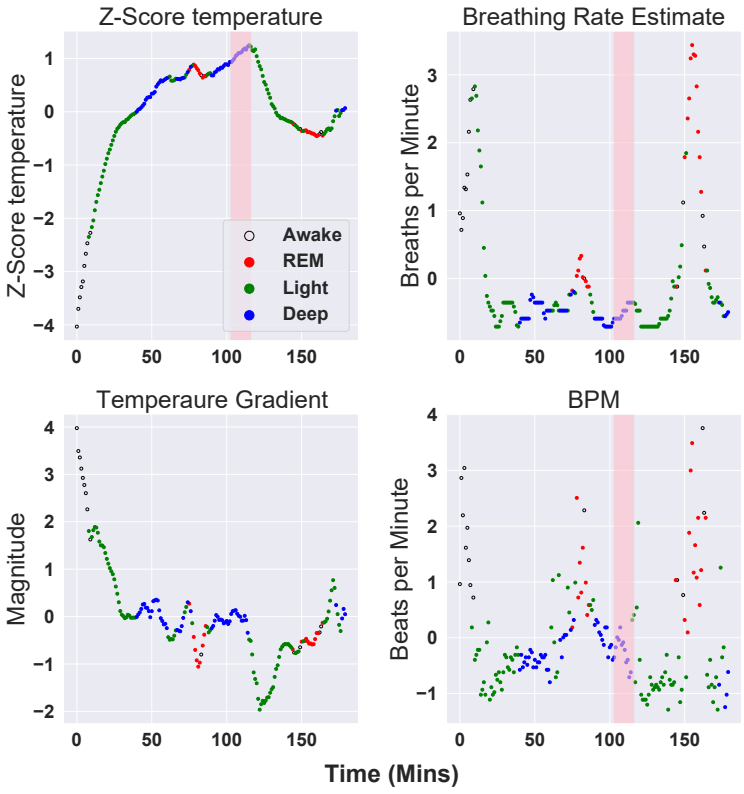


(2)

Figure 7.11: Poor REM performance for Fitbit 7.111 vs Automatic 7.112.



(1)



(2)

Figure 7.12: Poor deep performance for Fitbit 7.121 vs Automatic 7.122. Highlighting general under prediction of deep sleep for Fitbit.

7.2.3 Overall performance

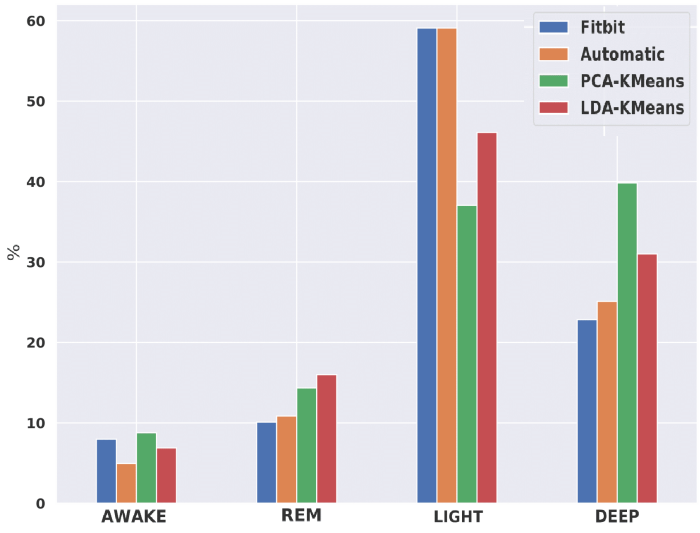
To assess the performance of the automated algorithm, figure 7.13 demonstrates the percentage of time spent in each sleep stage, averaged over 7 nights (the datasets from which the automatic method was devised) and the remaining 26 nights respectively. Figure 7.131 demonstrates all prediction approaches discussed in this chapter including: Fitbit (generated from data recored on wrist), automatic method (data from chest); PCA-KMeans (data from chest) and LDA-KMeans (data from chest). By nature, KMeans randomly assigns each cluster when applied. Hence, the initial predictions were saved and mapped to the appropriate stage relative to the previously discussed analysis. Comparing the percentage classifications over these 7 nights, it can be seen that there are minor disagreements between the Fitbit and the automatic method. These differences are significantly smaller than the differences noted between the Fitbit labels and the PCA-KMeans approach. As discussed, the LDA-KMeans approach generally improves this disagreement however as highlighted LDA uses the Fitbit labels to influence cluster separation (KMeans is not influenced by Fitbit labels however). While the clustering approach (PCA or LDA) clearly highlights the presence of the four sleep stages in the chest recorded data, there are lots of data points far from the cluster centres which do not get accurately classified. This was a key motivator for devising the automatic technique.

Figure 7.132 shows the performance of just the automatic algorithm when compared to the Fitbit labels on the reaming 26 nights of data. In general good agreement is present for estimating the percentage of time spent in each sleep stage, except for the Awake stage. To represent the potential inaccuracy of the Fitbit system, which is used as a baseline comparison, error bars have been added to figure 7.13. These are calculated based upon [63] in which for the classification of deep sleep the Fitbit was found to be 49% accurate compared to PSG, and light sleep to be 81% accurate. These values have been subtracted

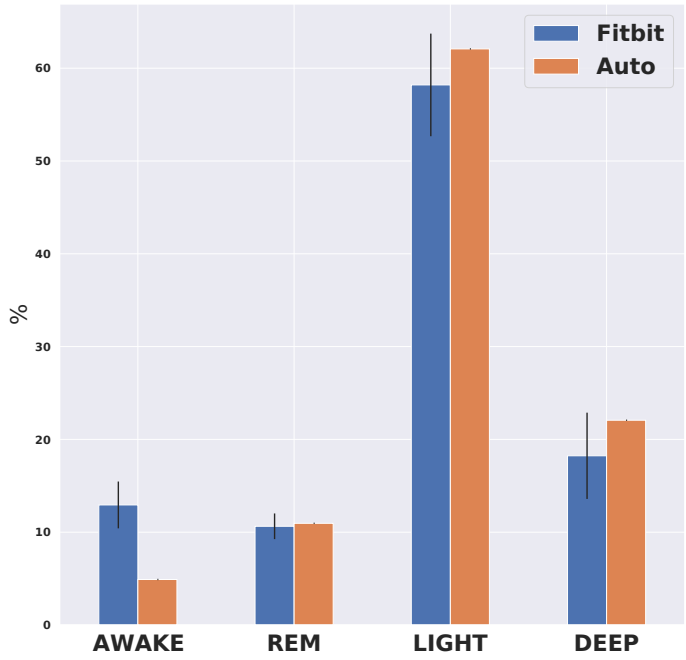
from 100% and multiplied by the average values produced from the Fitbit data to produce the errors bars. This provides an indirect comparison to PSG via the Fitbit smartwatch data. The poor performance noted on the Awake stage is likely due to the automatic method's poor ability to identify periods of short wakefulness during sleep (micro Awakes). Hence, overall, it under predicts the time spent Awake. It is theorised that the automatic method may over predict time spent in light sleep, therefore under predicting time spent awake. (It is noted that [63] featured 44 participants predominantly adult females. The study did not gender segregate the data so it is not possible to ascertain whether gender effects sleep stages.)

From this, it can be seen that the differences between the chest based predicitions and the Fitbit fall within the known error of the Fitbit device for all stages of sleep except Awake. Hence, its is concluded that the system and analysis approach demonstrate reasonable performance with regards to quantifying the structure of sleep over a 26 night period. Also, the system appears to be inline with what has been discussed with regards to the Fitbit under classification of deep sleep. From a sleep analysis point of view, this demonstrates that consumer grade sleep tracking is possible from chest worn sensors, opening opportunities for increasing the functionality of the many smart strap/patch devices which are now starting to emerge. The ability to record temperature and breathing rate from this location is a major advantage, as is motion-robust heart information using the ECG. With regards to sleep tracking in space, these results demonstrate the potential of limited sensor configurations to be used in microgravity. As discussed previously, large/uncomfortable sensor systems (such as PSG or EEG) are note suited for deployment into space. Attaching the necessary sensors to an elasticated, compact chest strap firstly attempts to ensure acceptable comfort for the user. Secondly, it maintains good sensor contact (where necessary) relatively to the user considering the absence of gravity. This concept does not

only apply to sleep stage analysis and could be applied to monitoring any other significant physiological concept that requires investigation in microgravity.



(1)



(2)

Figure 7.13: Mean percentage of time spent in each sleep stage. 7.131 7 nights of data, calculated by Fitbit, KMeans clustering and the automatic algorithm. 7.132 26 nights of data, comparing Fitbit reported values and to the automatic algorithm using only chest based data.

7.3 Conclusions

This chapter presents an approach to non-invasively perform sleep stage analysis via a chest worn sensor system that has been designed with the constraints of deployment into microgravity considered. The principal of the method was to analyse the sleep data monitored by the sensor platform in a reduced dimensionality coordinate space. The apparent clusters were then compared to the processed sensor data to identify important features and trends. Based on this analysis an automatic classification method has been devised and bench-marked against the sleep stage predictions produced by a Fitbit smartwatch. As demonstrated, the automated approach benefited from being situated in the chest region due to inclusion of extra sensing capabilities (compared to the Fitbit smartwatch) which in turn validates the use of such an approach in microgravity. Use of this approach in microgravity is beneficial for reasons as discussed but primarily it allows the size of such a system to be reduced and comfort for the user to be improved.

Chapter 8

Application of Previously Employed Feature Generation Methods to Multiple Participants

8.1 Introduction

The automatic sleep stage prediction method developed in chapter 7, was demonstrated to perform well when tested on a single participant. To further investigate, this chapter details the application of the sleep sensor system and associated algorithm to four additional subjects to understand the generalisation capabilities of the method. From this, it was determined that due to non desirable sensor responses relative to the extra participants, the automatic method was not capable of accurately determining the sleep stages for multiple users. Hence, a new sleep stage prediction method based on the feature generation methods used in chapter 6 was developed. This new method, which applies a clustering algorithm to the generated features, demonstrates better generalisation compared to the automatic

method when applied to the sleep data sampled from the additional participants. Also highlighted is the diversity of feature generation approaches considering the successful application to two different use cases, swim/sleep analysis. However, as stated, clustering algorithms are required here compared to chapter 6, which used classification methods. This is due to the lack of labels associated with the sleep data discussed in this chapter. In total, including chapter 7, five participants are considered during the sleep investigations of this work. Arguably, this is not statistically significant however does allow an initial understanding for the robustness of the devised algorithms.

8.2 Methods

8.2.1 Data acquisition process

Sleep datasets collected from 4 participants have been acquired using the sleep sensor system described by chapter 3, section 3.2.2. There were no electronic modifications made to the sensor system compared to the device used in chapter 7, nor any changes to the automatic sleep stage prediction algorithm.

All data presented has been obtained from 5 separate users, in accordance with relevant ethical approval processes in place at the time of testing.

8.3 Results & Discussion

When the automatic method was applied to the sleep data sampled from additional participants, it was determined to generate inaccurate sleep stage predictions as shown by figure 8.1. The performance of the automatic method here is questionable as the majority of samples are predicted as light sleep. According to literature, whilst asleep, a healthy human spends most time in the NREM stages [165, 166, 167]. The results demonstrated by

figure 8.1, indicate almost all of the time spent asleep is NREM. Practically non of which is comprised of deep sleep which did not seem realistic. Assessing the data highlights that the signal responses are different compared to the results observed in chapter 7. Firstly, the temperature recordings do not indicate similar magnitude changes compared to previous data sets. This issue was found to be the case for other participants, hence the temperature sensor was tested but not found to be broken/damaged. Therefore, it was assumed that the physical temperature changes during sleep are not consistent between individuals. The automatic algorithm relies on temperature measurements, particularly for deep sleep classifications. In addition, the previously seen clear correlation between increased breathing rate and BPM is less apparent. This could be due to multiple factors, possibly caused by physiological differences between individual subjects or a result of the high susceptibility to motion artefacts for breathing rate estimation from accelerometer data. The correlation between increasing BPM and breathing rate is crucial for classification of REM by the automatic algorithm.

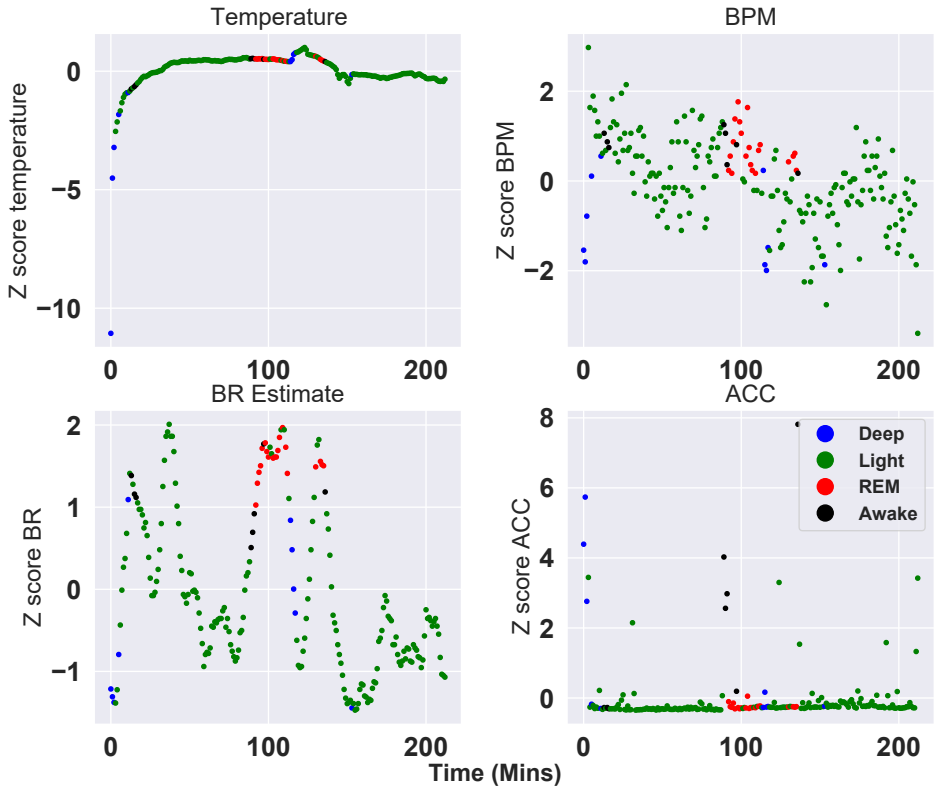


Figure 8.1: Sleep data, sampled for a different subject compared to the participant in chapter 7, processed via the automatic algorithm to predict sleep stages. Poor performance noted

This failure of the automatic algorithm, and in other cases, indicates the automatic method does not generalise well to other participants. Therefore, it is overfit to data sampled from the participant in chapter 7. Modifications to the automatic process were attempted however proved unsuccessful with regards to the general prediction of sleep stage. In attempt to resolve this, the feature generation methods employed in chapter 6 have been applied to the sleep data given the success achieved with regards to general swim stroke classification. The features generated are:

- Data Range

- Mean Delta
- Mean
- Max
- Kurtosis
- Skew
- Inter Qualtile Range (IQR)
- Standard Deviation
- Frequency Mean
- Frequency Max
- Frequency Kurtosis
- Frequency Skew
- Frequency IQR
- Frequency Standard Deviation

Where the automatic algorithm interprets the processed physiological information (BPM, breathing rate etc), the feature generation techniques are applied to the raw data. Specifically ECG and accelerometer x/y/z axis data. Temperature data is not included, this is done due to the inconsistent response of this sensor as discussed above. A crucial difference between the feature generation approach used in this chapter, compared to the automatic approach used in chapter 7, is that the automatic method processes data that has been averaged over a minute. The methods developed in this chapter focus on the raw 10 second

ECG/accelerometer samples recorded by the sleep sensor system. However, subsequent analysis in this chapter uses results averaged over a minute to allow comparison against the process physiological data; this process is detailed below.

As mentioned above, there are no labels associated to the generated features. Hence unsupervised methods are required. Post feature generation, the resultant dataset has 56 features. As before, PCA (see chapter 2, section 2.3.1) is applied to this data, this is done in attempt to improve the results of the subsequently applied clustering algorithms. Figure 8.2 demonstrates how much of the total variance is explained compared to the number of principal components. As clustering is the only sleep prediction method used in this chapter, it is important to maximise the explained variance whilst attempting to increase separability. Hence, 32 principal components are selected for analysis given the flattening out of the explained variance demonstrated by figure 8.2 around this mark.

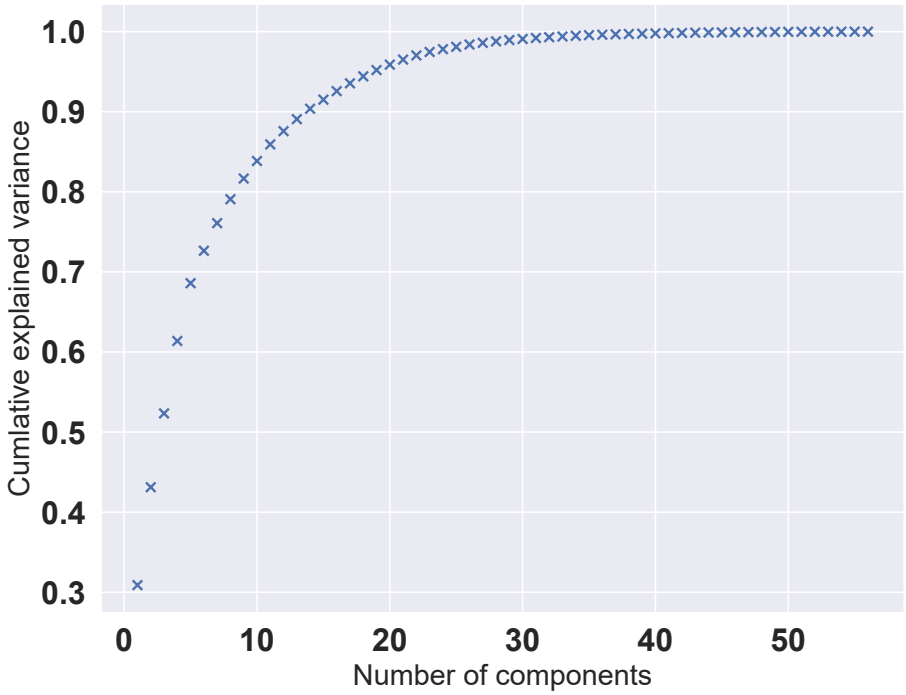


Figure 8.2: Cumulative variance explained by each principal component.

Figure 8.3 details the first 2 principal components for this representation of the dataset. There appears to be some concentrations of data points hence KMeans (number of clusters set as five - see chapter 2, section 2.3.1 for refernce) was applied as demonstrated by the colour coding. Note, clustering is applied to the 32 feature dataset (processed via PCA), two dimensions are displayed by figure 8.3 for simplicity. KMeans is configured to identify five clusters, instead of four clusters (matching the number of sleep stages) as before. This was done, as mentioned above, because the focus of this clustering is to overlay the results onto the processed physiological information that is average over a minute. Thus, the clustering results must also be averaged over a minute which was implemented using the mode. Experimentation determined locating five clusters at this initial stage, ensures four clusters are present post data averaging. In some cases, if five clusters were present post

averaging, for simplicity the minority cluster was automatically assigned as the second smallest cluster. It is possible that this may result in errors, however post averaging the minority class is often minimal in size hence the detrimental affect is not significant.

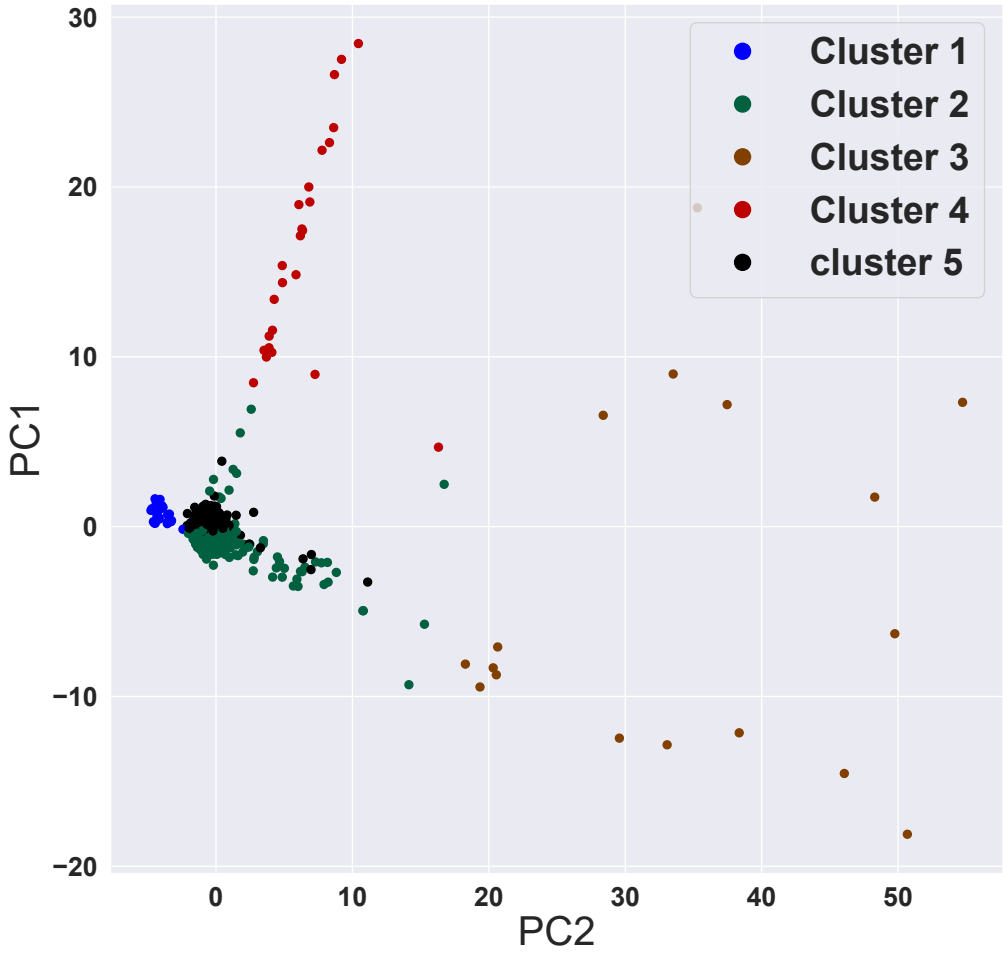


Figure 8.3: KMeans clustering (k=5) applied to the first 32 principal components. Two principal components shown for simplicity.

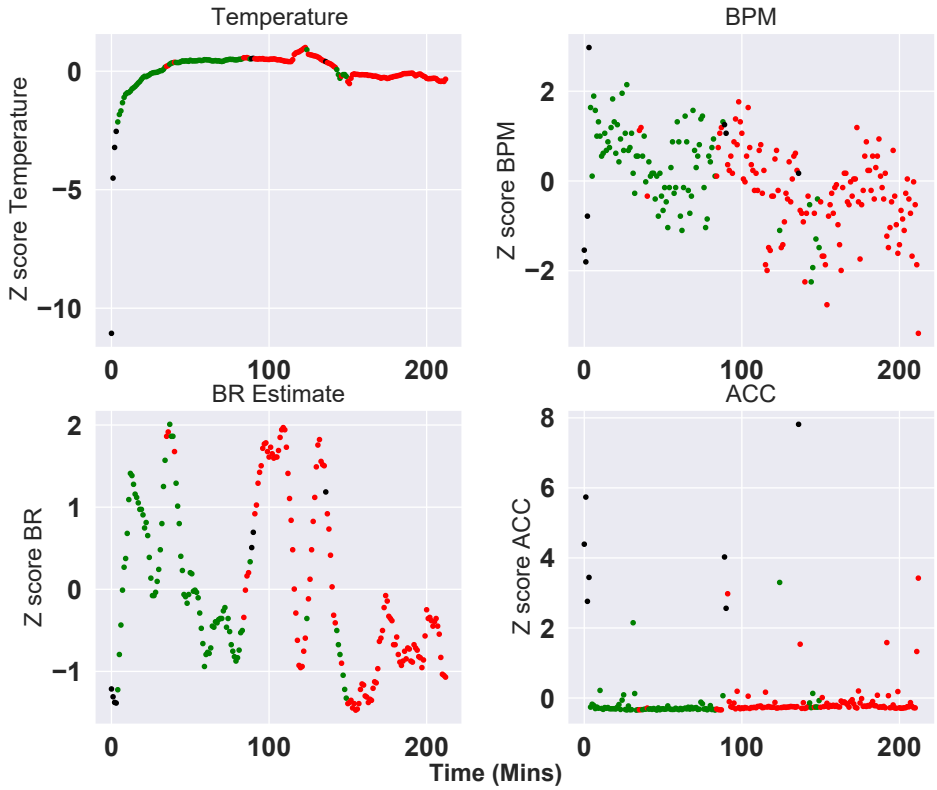


Figure 8.4: Averaged KMeans-PCA clustering results applied to the processed physiological data. Slightly improved performance, still not optimal.

Figure 8.4 demonstrates the processed physiological information, colour coded according to the averaged clustering results. Clustering the PCA data does not appear to offer significant improvement compared to the automatic method when analysis is performed against signal data. It is possible that the red cluster may somewhat correlate to REM sleep, however it appears no realistic deep sleep stages are identified. It was speculated that the relationship between the generated features may not be linear. PCA by nature can only increase the linear separability of a dataset, hence it is possible that important relationships are not identified if the data is non linear. Thus, an autoencoder approach (see chapter 2, section 2.4.4), given the non linear capabilities of neural networks was in-

vestigated to verify this. As discussed in chapter 2, an autoencoder is comprised of an encoder (structure demonstrated by figure 8.5, layers detailed by table 8.1) and a decoder (structure demonstrated by figure 8.6, layers detailed by table 8.2). These two components are linked during training, but split once training is complete and the encoder section is used to generate the encoded data.

As shown by table 8.1, the output layer of the encoder has 5 neurons, this is chosen through experimentation which indicated that clustering in 5 dimensions gave the best results. The output layer of the decoder has 56 neurons as demonstrated by table 8.2 to match the dimensionality of the original dataset.

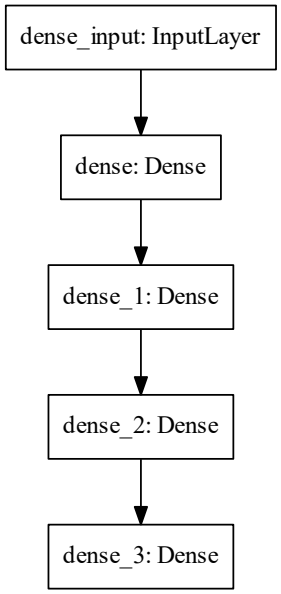


Figure 8.5: Encoder section structure.

Table 8.1: decoder layer(s) configuration.

Layer	Number of neurons	Activation
1	40	ELU
2	20	ELU
3	10	ELU
4 (Encoder output)	5	ELU

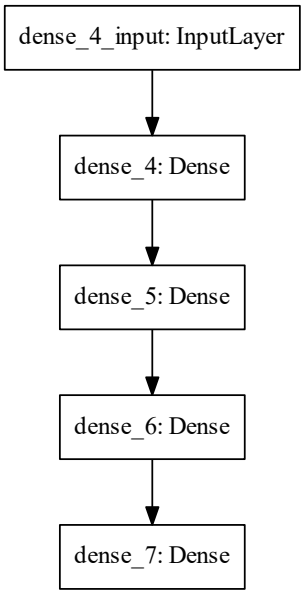


Figure 8.6: Decoder section structure.

Table 8.2: Decoder layer(s) configuration.

Layer	Number of neurons	Activation
1	10	ELU
2	20	ELU
3	40	ELU
4 (Decoder output)	56	ELU

Figure 8.7 demonstrates the first two dimensions of the encoded dataset to which KMeans has been applied (number of clusters set at 5 as before). Figure 8.8 overlays the results of this clustering, averaged over a minute, onto the processed physiological data (same data as figure 8.1/8.4). Based on the analysis performed in chapter 7, the clusters have been correlated to the sleep stages as shown by the legend on this figure. The results here seem more likely compared to what was generated by figures 8.1/8.4. Some concerns remain, namely the possibility that light sleep might be underpredicited here which is theorised based on visual analysis. However, crucially, this method generalises well when tested on the other subjects which is promising considering the automatic algorithms lack of flexibility. To further understand this new approach, an overall performance, as with chapter 7 was performed comparing the predicted results to the sleep stage classifications made by a Fitbit smartwatch. This was done using the 28 datasets, all sampled from one subject, analysed in chapter 7 to ensure fair comparison. Figure 8.9 demonstrates a histogram detailing percentage time spent in each sleep stage including error bars applied to the Fitbit data which relate to the smartwatch bias compared to PSG.

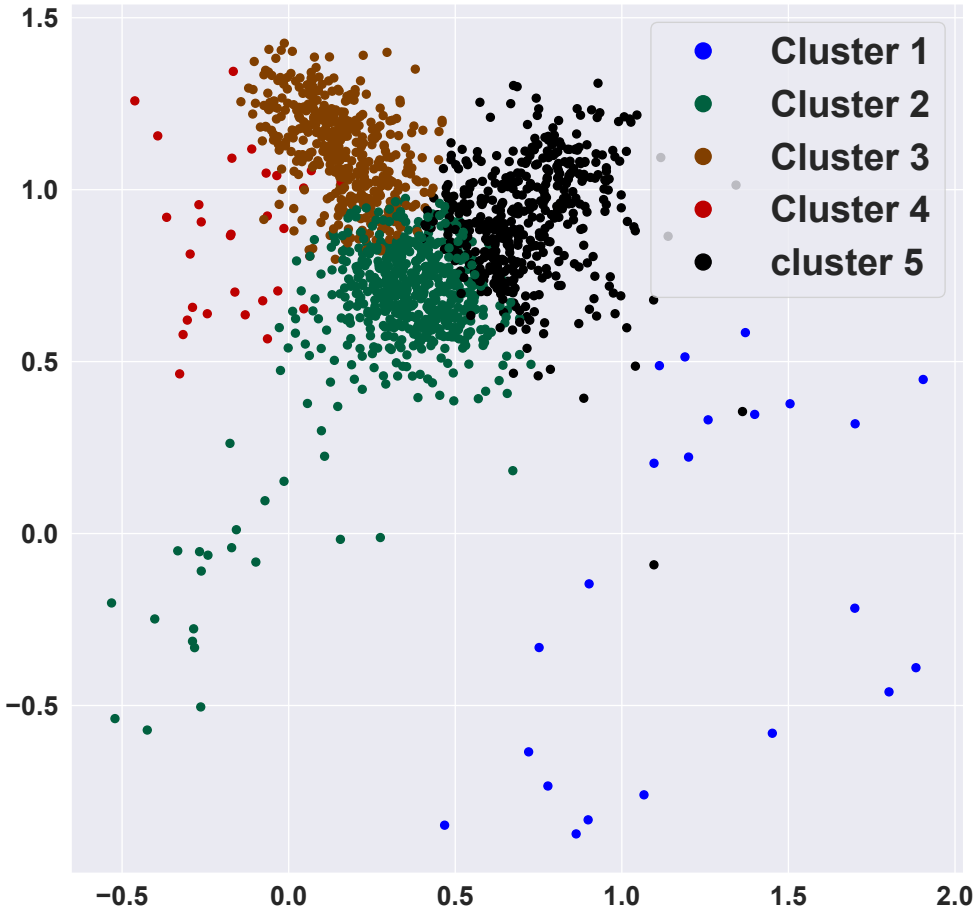


Figure 8.7: KMeans clustering (k=5) applied in three dimensions. Two dimensions components shown for simplicity.

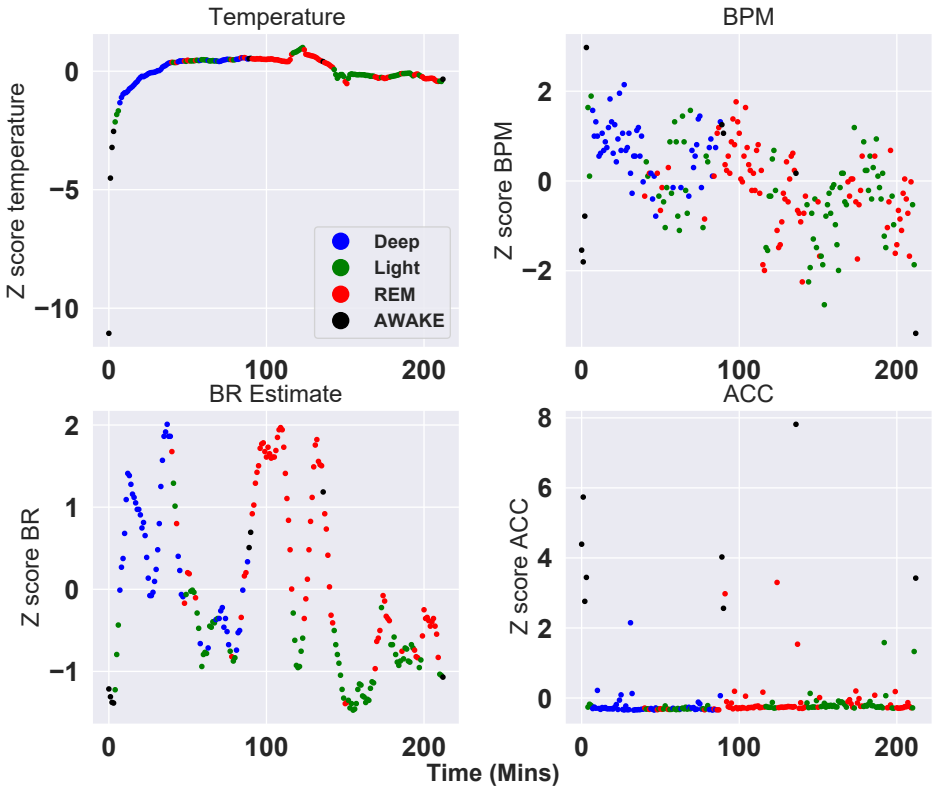


Figure 8.8: Averaged KMeans-Encoded data clustering results applied to the processed physiological data. Sleep stages indicated according to assessment of processed physiological data.

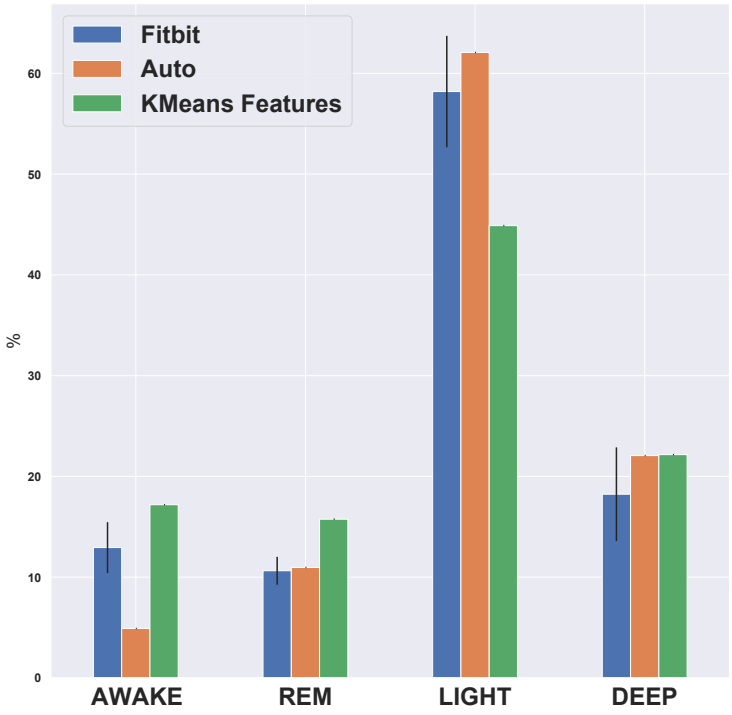


Figure 8.9: Comparison of percentage time spent in each sleep stage from predictions made by Fitbit, automatic and encoder clustering methods applied to same data demonstrated by figure 7.9

As demonstrated by figure 8.9, KMeans clustering applied to the encoded features predicts less light sleep relative to the Fitbit and automatic method. This is mirrored by an increased prediction of REM and awake sleep and an overall lack of agreement with error bars as shown. To investigate this, comparison was made to other literature sources which state REM sleep should constitute approximately 20-25% of sleep versus NREM which accounts for 75-80% [165, 166, 167]. In addition to this, PSG datasets accessed online (specifically [168]) confirm the ratio between NREM and REM sleep for which 82% is NREM and 18% is REM. This study involved 151 individuals. Considering this, for time spent asleep, the Fitbit predicts approximately 12% of time spent in REM and 82% time spent in NREM. Also, the automatic predicts approximately 12% REM, 82% NREM.

Compared to the KMeans method which predicts 20% REM and 80% NREM, aligned with the discussed literature values. This contradicts previously demonstrated error bars, which indicate good performance relative to PSG as discussed in chapter 7. Ultimately, because much of this work was performed during the COVID-19 pandemic, access to PSG was not possible which meant a direct comparison to the gold standard for sleep stage classification could not be performed. Comparison could not be made to the datasets accessed online (as discussed above), as this data did not include information measured from the same sensors compared to the device developed in this thesis. ECG was the only like sensor featured hence there was no basis for a direct comparison. Although the Fitbit has been compared to PSG, and the techniques developed here compared to the Fitbit, this does not directly relate the performance.

To further investigate and attempt to understand the contradiction presented above, the sleep data sampled from the additional four participants (3 nights from subject one, 1 night from subject two, 2 nights from subject three and 2 nights from subject 4), was analysed via the automatic algorithm and KMeans method devised in this chapter. The absence of Fitbit labels for this data meant analysis was performed using the known NREM/REM composition of sleep as discussed above. Table 8.3 compares the average percentage time, spent in NREM/REM, predicted by both methods per subject. As demonstrated, the automatic algorithm consistently over predicts NREM sleep relative to REM, whereas the KMeans method better agrees with the literature. This indicates that the KMeans approach is more suited to identifying the sleep stages considering the evidence available. In reality, all individuals sleep differently, even in the absence of a sleep disorder(s). Given that the analysis performed here is on a small sample with limited nights of sleep recorded, the averages calculated may not be representative of an individuals sleep in general. Regardless, the KMeans method better agrees with known values indicating improved generalisation

and performance of this technique relative to the automatic method.

Table 8.3: Averaged percentage time predicted per sleep subject

Subject	Method	Stage	Percentage time (%)
1	Automatic	NREM	93
		REM	7
	KMeans	NREM	69
		REM	31
2	Automatic	NREM	93
		REM	7
	KMeans	NREM	80
		REM	20
3	Automatic	NREM	89
		REM	11
	KMeans	NREM	67
		REM	33
4	Automatic	NREM	84
		REM	17
	KMeans	NREM	70
		REM	30

8.4 Conclusions

This chapter investigates application of the sleep sensor system and sleep stage prediction methods applied to other participants, compared to the single subject featured in chapter

7. The automatic algorithm (developed in chapter 7) was found not to generalise well to the additional participants. Hence, a new sleep prediction method was devised involving generating multiple features from the raw signal data (as with chapter 6), encoding this data via application of a trained encoder, and then applying KMeans clustering to this encoded information. Use of encoder prior to application of KMeans performs better than PCA possibly due to non linear relationships within the data. The clustering results have been correlated to the sleep stages so that analysis similar to that performed in chapter 7 can be performed. This demonstrates that the clustering of the encoded sleep data performs well for awake and deep however under predicts light sleep/over predicts REM. However, the methods developed in this chapter can be applied to multiple users instead of overfitting to a single person. In addition, the methods introduced here are unsupervised meaning that no training data is required to generate sleep stage predictions.

Chapter 9

Conclusion

To conclude, this work details the construction of a wearable sensor platform capable of deployment into the extreme environments considered during this thesis, which are microgravity and underwater. Focus has been directed towards the monitoring of physiological data and the subsequent processing of this information into metrics relative to the given scenario. Specifically, for the microgravity environment, data is processed into sleep stage information whereas during the underwater investigation swim analysis metrics are generated. Signal/data processing techniques are favoured instead of the selection/fabrication of specialised senors, this is done in attempt to minimise cost and ensure easy deployment. The two environments considered present unique difficulties with regards to the challenges associated with sensing in extreme environments, namely whether an environment limits sensor performance or if human performance is inhibited. The distinction here is important and appropriate considerations must be made to account for these factors. However, this work demonstrates the extension of certain methods/strategies to both types of environment.

The results shown in this thesis indicate generally good performance of the constructed

sensor system. The sensors used are able to monitor both small (sleep stages) and large (swimming) signal variations and the associated algorithms can interpret this information. The results noted during the underwater analysis investigations (both ECG and combined sensors) indicate the developed algorithms are good indicators of the relevant metrics. Techniques such as the use of artefact replication to ensure machine learning algorithms can recognise such noise/distortions, has applications in many other use cases besides underwater analysis. Difficulties arose during the instigation of sleep stage prediction methods, in particular when investigating general application to multiple participants. However, using methods applied during the swim analysis work, feature generation in particular, appears to alleviate this issue hence creating a link between the two use cases. The objectives specified for this work have been achieved. The constructed sensor system was built for under the specified cost target and the developed algorithms generally achieve the target classification/regression scores specified by the objectives for this thesis. Particularly the devised under water BPM prediction method which has been published. In addition the swim stroke classification algorithm performs well. Arguably, examples in literature perform as well if not better however such cases require the use of multiple IMUs instead of a single accelerometer.

The work demonstrated in this thesis, for both environments discussed, has extensions beyond what is introduced here. As the expansion of wearable sensor technology progresses and the requirement for sensing in extreme environments remains, new methods will be required to ensure both sensor and algorithmic functionality in such situations. This work has demonstrated that there are inexpensive options and design choices available for the use of sensors in extreme environments. Future work shall consider publishing work on sleep stage classification, specifically involving feature generation methods.

Bibliography

[1] T. Beckingham, J. Spencer, and K. McKay, "Sleep Monitoring in Microgravity," *International Aeronautical Conferencer*, 2019.

[2] T. Beckingham, J. Spencer, and K. Mckay, "CNN-RNN architecture to calculate BPM from underwater ECG samples," *Applied Intelligence*, 2023.

[3] X. L. Yang, G. Z. Liu, Y. H. Tong, H. Yan, Z. Xu, Q. Chen, X. Liu, H. H. Zhang, H. B. Wang, and S. H. Tan, "The history, hotspots, and trends of electrocardiogram," *Journal of Geriatric Cardiology*, vol. 12, no. 4, pp. 448–456, 2015.

[4] M. AlGhatrif and J. Lindsay, "A brief review: history to understand fundamentals of electrocardiography," *Journal of Community Hospital Internal Medicine Perspectives*, vol. 2, no. 1, p. 14383, 2012.

[5] S. Patel, H. Park, P. Bonato, L. Chan, and M. Rodgers, "A review of wearable sensors and systems with application in rehabilitation," *Journal of NeuroEngineering and Rehabilitation*, vol. 9, no. 1, p. 21, 2012.

[6] Fitbit, "How does my Fitbit device calculate my daily activity?." https://help.fitbit.com/articles/en$_US$/$Help_article$/1141.htm.

[7] G. Aroganam, N. Manivannan, and D. Harrison, "Review on Wearable Technology Sensors Used in Consumer Sport Applications," *Sensors (Basel, Switzerland)*, vol. 19, no. 9, 2019.

[8] Analog Devices, "AD8232 Single-Lead ECG," *Data Sheet*, pp. 1–28, 2013.

[9] NHS, "How do I check my pulse," pp. 1–2, 2021.

[10] "Understanding an ECG — ECG Interpretation — Geeky Medics." https://geekymedics.com/understanding-an-ecg/.

[11] "What is the difference between an ECG, EEG, EMG and EOG?." https://www.sensortips.com/featured/what-is-the-difference-between-an-ecg-eeg-emg-and-eog/.

[12] "Electromyography (EMG) - Mayo Clinic." https://www.mayoclinic.org/tests-procedures/emg/about/pac-20393913.

[13] "Electroencephalogram (EEG) - NHS." https://www.nhs.uk/conditions/electroencephalogram/.

[14] D. W. Carley and S. S. Farabi, "Physiology of sleep," *Diabetes Spectr.*, vol. 29, no. 1, pp. 5–9, 2016.

[15] A. Affanni, "Wireless sensors system for stress detection by means of ecg and eda acquisition," *Sensors (Switzerland)*, vol. 20, 4 2020.

[16] H. N. Denisse Castaneda, Aibhlin Esparza, Mohammad Ghamari, Cinna Soltanpur, "A review on wearable photoplethysmography sensors and their potential future applications in health care," *Physiology & behavior*, vol. 176, no. 1, pp. 100–106, 2016.

[17] D. T. Weiler, S. O. Villajuan, L. Edkins, S. Cleary, and J. J. Saleem, "Wearable heart rate monitor technology accuracy in research: A comparative study between PPG and ECG

technology," *Proc. Human Factors and Ergonomics Society*, vol. 61, no. 1, pp. 1292–1296, 2017.

[18] Z. Zhang, J. Zheng, H. Wu, W. Wang, B. Wang, and H. Liu, "Development of a respiratory inductive plethysmography module supporting multiple sensors for wearable systems," *Sensors (Switzerland)*, vol. 12, no. 10, pp. 13167–13184, 2012.

[19] G. Z. Liu, Y. W. Guo, Q. S. Zhu, B. Y. Huang, and L. Wang, "Estimation of respiration rate from three-dimensional acceleration data based on body sensor network.," *Telemedicine journal and e-health : the official journal of the American Telemedicine Association*, vol. 17, no. 9, pp. 705–711, 2011.

[20] S. K. Mukhopadhyay and S. Krishnan, "Robust identification of qrs-complexes in electrocardiogram signals using a combination of interval and trigonometric threshold values," *Biomedical Signal Processing and Control*, vol. 61, 8 2020.

[21] H. Hymczak, A. Gołąb, K. Mendrala, D. Plicner, T. Darocha, P. Podsiadło, D. Hudziak, R. Gocoł, and S. Kosiński, "Core temperature measurement—principles of correct measurement, problems, and complications," *International Journal of Environmental Research and Public Health*, vol. 18, no. 20, 2021.

[22] A. P. Welles, X. Xu, W. R. Santee, D. P. Looney, M. J. Buller, A. W. Potter, and R. W. Hoyt, "Estimation of core body temperature from skin temperature , heat flux , and heart rate using a Kalman filter," *Computers in Biology and Medicine*, vol. 99, no. May, pp. 1–6, 2018.

[23] A. Jubran, "Pulse oximetry," *Critical Care*, vol. 19, no. 1, pp. 1–7, 2015.

[24] R. Giannetti, J. P. Silveira, M. L. Dotor, D. Golmayo, P. Martín, F. Miguel-Tobal, A. Bilbao, and S. M. López-Silva, "Oxygen saturation measurements in athletes attaining max-

imal exertion conditions," *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, vol. 1, no. June 2014, pp. 740–744, 2004.

[25] R. T. Li, S. R. Kling, M. J. Salata, S. A. Cupp, J. Sheehan, and J. E. Voos, "Wearable Performance Devices in Sports Medicine," *Sports Health*, vol. 8, no. 1, pp. 74–78, 2016.

[26] I. Arun Faisal, T. Waluyo Purboyo, and A. Siswo Raharjo Ansori, "A Review of Accelerometer Sensor and Gyroscope Sensor in IMU Sensors on Motion Capture," *Journal of Engineering and Applied Sciences*, vol. 15, no. 3, pp. 826–829, 2019.

[27] Bosch Sensortec, "BNO055 Intelligent 9-axis absolute orientation sensor," no. November, p. 105, 2014.

[28] "Insomnia - NHS." https://www.nhs.uk/conditions/insomnia/.

[29] "Narcolepsy - NHS." https://www.nhs.uk/conditions/narcolepsy/.

[30] "Sleep apnoea - NHS." https://www.nhs.uk/conditions/sleep-apnoea/.

[31] "Polysomnography (sleep study) - Mayo Clinic." https://www.mayoclinic.org/tests-procedures/polysomnography/about/pac-20394877.

[32] P. Anderer, G. Gruber, S. Parapatics, P. S. Fau, E. Loretz, M. Boeck, G. Kloesch, E. Heller, A. Schmidt, H. Danker-Hopfe, B. Saletu, J. Zeitlhofer, and G. Dorffner, "Sleep classification according to AASM and Rechtschaffen & Kales: effects on sleep scoring parameters," *Sleep*, vol. 32, no. 2, pp. 139–149, 2007.

[33] Institute of Medicine (US) Committee on Sleep Medicine and Research, "Sleep disorders and sleep deprivation: An unmet public health problem," in *The National Academies Collection: Reports funded by National Institutes of Health* (H. R. Colten and B. M. Altevogt, eds.), Washington (DC): National Academies Press, 2008.

[34] R. V. Rial, M. C. Nicolau, M. Akaa, A. Gamundõâ, and J. Gonza, "Why we sleep: the evolutionary pathway to the mammalian sleep," *Prog. Neurobiol.*, vol. 62, no. 4, pp. 379–406, 2000.

[35] J. Zabalza, J. Ren, J. Zheng, H. Zhao, C. Qing, Z. Yang, P. Du, and S. Marshall, "Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging," *Neurocomputing*, vol. 185, no. 1, pp. 1–10, 2016.

[36] S. Z. Assefa, M. Diaz-Abad, E. M. Wickwire, and S. M. Scharf, "The functions of sleep," *AIMS Neurosci.*, vol. 2, no. 3, pp. 155–171, 2015.

[37] A. Rechtschaffen, "Current perspectives on the function of sleep," *Perspect. Biol. Med.*, vol. 41, no. 3, pp. 359–390, 1998.

[38] G. Läckgren, H. Jerker, A. Stenberg, K. Hjälmås, T. Nevéus, and T. Tuvemo, "Enuresis—background and treatment," *Scand. J. Urol.*, vol. 34, no. 206, pp. 1–44, 2002.

[39] E. C. Harding, N. P. Franks, and W. Wisden, "The temperature dependence of sleep," *Frontiers in Neuroscience*, vol. 13, no. APR, pp. 1–16, 2019.

[40] D. McGinty and R. Szymusiak, "Brain structures and mechanisms involved in the generation of NREM sleep: Focus on the preoptic hypothalamus," *Sleep Med. Rev.*, vol. 5, no. 4, pp. 323–342, 2001.

[41] T. Penzel, K. Stephan, S. Kubicki, and W. M. Herrmann, "Integrated sleep analysis, with emphasis on automatic methods," *Epilepsy Res. Suppl.*, vol. 2, no. 1, pp. 177–204, 1991.

[42] A. Malhotra, M. Younes, S. T. Kuna, R. Benca, C. A. Kushida, J. Walsh, A. Hanlon, B. Staley, A. I. Pack, and G. W. Pien, "Performance of an automated polysomnography scoring system versus computer-assisted manual scoring," *Sleep*, vol. 36, no. 4, pp. 573–582, 2013.

[43] S. F. Liang, C. E. Kuo, Y. H. Hu, and Y. S. Cheng, "A rule-based automatic sleep staging method," *J. Neurosci. Meth.*, vol. 205, no. 1, pp. 169–176, 2012.

[44] R. Bogue, "Sensors for extreme environments," *Sensor Review*, vol. 32, no. 4, pp. 267–272, 2012.

[45] "PCB 3501A1260KG - 540-MEMS high shock sensor." https://www.dollenmeier.ch/en/piezoresistive-high-shock-sensors/538-pcb-3501a1260kg.html.

[46] R. A. Matula, "Electrical resistivity of copper, gold, palladium, and silver," *Journal of Physical and Chemical Reference Data*, vol. 8, no. 4, pp. 1147–1298, 1979.

[47] P. Horowitz and W. Hill, *The art of electronics; 3rd ed.* Cambridge: Cambridge University Press, 2015.

[48] M. Riordan and L. Hoddeson, "Origins of the pn junction," *IEEE Spectrum*, vol. 34, no. 6, pp. 46–51, 1997.

[49] P. Chakravorty, "A Modified General Diode Equation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 1, no. c, pp. 1–5, 2021.

[50] H. A. Abdulqader, "BJT Fabrication Using Excimer Laser Assisted Spin-on Doping Technique," *Communication*, vol. 6, pp. 197–202, 2008.

[51] N. L. Jeon, J. Hu, G. M. Whitesides, M. K. Erhardt, and R. G. Nuzzo, "Fabrication of silicon MOSFETs using soft lithography," *Advanced Materials*, vol. 10, no. 17, pp. 1466–1469, 1998.

[52] S. Kalra, "Effect of temperature dependence on performance of digital CMOS circuit technologies," *2013 International Conference on Signal Processing and Communication, ICSC 2013*, pp. 392–395, 2013.

[53] K. I. Nuttall, O. Buiu, and V. V. Obreja, "Surface leakage current related failure of power silicon devices operated at high junction temperature," *Microelectronics Reliability*, vol. 43, no. 9-11, pp. 1913–1918, 2003.

[54] R. Mishra, "The Temperature Ratings Of Electronic Parts — Electronics Cooling." https://www.electronics-cooling.com/2004/02/the-temperature-ratings-of-electronic-parts/.

[55] R. L. Sumwalt, "Enhancing flight-crew monitoring skills can increase corporate aviation safety," *Proceedings of the Corporate Aviation Safety Seminar*, vol. 49, pp. 197–203, 2004.

[56] A. Cuc, V. Giurgiutiu, S. Joshi, and Z. Tidwell, "Structural health monitoring with piezoelectric wafer active sensors for space applications," *AIAA Journal*, vol. 45, no. 12, pp. 2838–2850, 2007.

[57] Y. Wang, D. Liu, D. Xu, B. Wu, F. Wang, and X. Wu, "On-orbit sleep problems of astronauts and countermeasures," *Mil. Med. Res.*, vol. 5, no. 1, pp. 1–12, 2018.

[58] G. K. Prisk, "Sleep and respiration in microgravity.," *Neuroscience news*, vol. 1, no. 5, pp. 39–45, 1998.

[59] S. R. Pandi-Perumal and A. A. Gonfalone, "Sleep in space as a new medical frontier: The challenge of preserving normal sleep in the abnormal environment of space missions," *Sleep Science*, vol. 9, no. 1, pp. 1–4, 2016.

[60] B. Rusak and I. Zucker, "Neural Regulation of Circadian Rhythms," *Physiological Reviews*, vol. 59, no. 3, p. 958, 1979.

[61] D.-J. Dijk, D. F. Neri, J. K. Wyatt, J. M. Ronda, E. Riel, A. Ritz-De Cecco, R. J. Hughes, A. R. Elliott, G. K. Prisk, J. B. West, and C. A. Czeisler, "Sleep, performance, circadian rhythms, and light-dark cycles during two space shuttle flights," *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, vol. 281, no. 5, pp. R1647–R1664, 2017.

[62] F. W. Boltz, "Orbital motion under continuous radial thrust," *Journal of Guidance, Control, and Dynamics*, vol. 14, no. 3, pp. 667–670, 1991.

[63] M. de Zambotti, A. Goldstone, S. Claudatos, I. M. Colrain, and F. C. Baker, "A validation study of Fitbit Charge 2™ compared with polysomnography in adults," *Chronobiol. Int.*, vol. 35, no. 4, pp. 465–476, 2018.

[64] "EMFIT QS 2 WiFi - Contact-free sleep tracker for personal use - Emfit Ltd (Finland)." https://emfit.com/product/emfit-qs-2-wifi-contact-free-sleep-tracker-for-personal-use/.

[65] S. P. Preejith, A. Jeelani, P. Maniyar, J. Joseph, and M. Sivaprakasam, "Accelerometer based system for continuous respiratory rate monitoring," in *IEEE MeMeA*, (Rochester), pp. 171–176, 2017.

[66] X. Xu, A. J. Karis, M. J. Buller, and W. R. Santee, "Relationship between core temperature, skin temperature, and heat flux during exercise in heat," *Eur. J. Appl. Physiol.*, vol. 113, no. 9, pp. 2381–2389, 2013.

[67] "Conductivity (Electrical Conductance) and Water — U.S. Geological Survey." https://www.usgs.gov/special-topics/water-science-school/science/conductivity-electrical-conductance-and-water.

[68] A. Wolf and B. Raton, "Electrical Conductivity of Aqueous Solutions," *CRC Press*, pp. 1–6, 1989.

[69] J. J. Barron and C. Ashton, "Conductivity Measurement - A REAGECON TECHNICAL PAPER," *A Practical Guide to Accurate Conductivity Measurements*, vol. 7, no. 3, pp. 1–5, 2013.

[70] Mary River Catchment Coordinating Committee, "Water quality standards," p. 500, 2013.

[71] J. Veerman, "The effect of the NaCl bulk concentration on the resistance of ion exchange membranes-measuring and modeling," *Energies*, vol. 13, no. 8, 2020.

[72] C. D. o. O. P. Issue, "CONDUCTIVITY MEASUREMENT IN HIGH PURITY WATER SAMPLES below 10 µ SIEMENS / cm High Purity Water Resistivity / Conductivity Measurement," *IC COntrols of Process Water Analysis Equipments*, no. 4, 2012.

[73] R. Avram, G. H. Tison, K. Aschbacher, P. Kuhar, E. Vittinghoff, M. Butzner, R. Runge, N. Wu, M. J. Pletcher, G. M. Marcus, and J. Olgin, "Real-world heart rate norms in the Health eHeart study," *npj Digital Medicine*, vol. 2, no. 1, 2019.

[74] B. A. Reyes, H. F. Posada-Quintero, J. R. Bales, A. L. Clement, G. D. Pins, A. Swiston, J. Riistama, J. P. Florian, B. Shykoff, M. Qin, and K. H. Chon, "Novel electrodes for underwater ECG monitoring," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 6, pp. 1863–1876, 2014.

[75] B. A. Reyes, H. F. Posada-Quintero, J. R. Bales, and K. H. Chon, "Performance evaluation of carbon black based electrodes for underwater ECG monitoring," *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2014*, pp. 1691–1694, 2014.

[76] F. Stauffer, M. Thielen, C. Sauter, S. Chardonnens, S. Bachmann, K. Tybrandt, C. Peters, C. Hierold, and J. Vörös, "Skin conformal polymer electrodes for clinical ecg and eeg recordings," *Advanced Healthcare Materials*, vol. 7, 4 2018.

[77] S. Gradl, T. Cibis, J. Lauber, R. Richer, R. Rybalko, N. Pfeiffer, H. Leutheuser, M. Wirth, V. von Tscharner, and B. M. Eskofier, "Wearable current-based ECG monitoring system with non-insulated electrodes for underwater application," *Applied Sciences (Switzerland)*, vol. 7, no. 12, 2017.

[78] R. Vullings, M. J. Rooijakkers, and M. Mischi, "Latency variable source separation for heart rate detection in low-quality ECG signals," *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pp. 6935–6938, 2011.

[79] M. A. Kobat, O. Karaca, P. D. Barua, and S. Dogan, "Prismatoidpatnet54: An accurate ecg signal classification model using prismatoid pattern-based learning architecture," *Symmetry*, vol. 13, no. 10, pp. 1–13, 2021.

[80] M. Baygin, T. Tuncer, S. Dogan, R. S. Tan, and U. R. Acharya, "Automated arrhythmia detection with homeomorphically irreducible tree technique using more than 10,000 individual subject ECG records," *Information Sciences*, vol. 575, pp. 323–337, 2021.

[81] H. Chen and K. Maharatna, "An Automatic R-peak Detection Method Based on Hierarchical Clustering," *BioCAS 2019 - Biomedical Circuits and Systems Conference, Proceedings*, pp. 0–3, 2019.

[82] A. Darmawahyuni, S. Nurmaini, M. N. Rachmatullah, B. Tutuko, A. I. Sapitri, F. Firdaus, A. Fansyuri, and A. Predyansyah, "Deep learning-based electrocardiogram rhythm and beat features for heart abnormality classification," *PeerJ Computer Science*, vol. 8, 2022.

[83] Garmin, "HRM-Swim™," pp. 3–5, 2021.

[84] B. Askarian, K. Jung, and J. W. Chong, "Monitoring of heart rate from photoplethysmographic signals using a Samsung Galaxy Note8 in underwater environments," *Sensors (Switzerland)*, vol. 19, no. 13, 2019.

[85] M. S. Pan, K. C. Huang, T. H. Lu, and Z. Y. Lin, "Using accelerometer for counting and identifying swimming strokes," *Pervasive and Mobile Computing*, vol. 31, pp. 37–49, 2016.

[86] M. Estivalet and P. Brisson, "ACCELEROMETER PROFILE RECOGNITION OF SWIMMING STROKES," *The Engineering of Sport 7*, no. August, pp. 0–6, 2008.

[87] Z. Zhang, D. Xu, Z. Zhou, J. Mai, Z. He, and Q. Wang, "IMU-based underwater sensing system for swimming stroke classification and motion analysis," *2017 IEEE International Conference on Cyborg and Bionic Systems, CBS 2017*, vol. 2018-Janua, pp. 268–272, 2017.

[88] "Buy Apple Watch Series 8 - Apple (UK)." https://www.apple.com/uk/shop/buy-watch/apple-watch.

[89] "Garmin Swim™ 2 — Swim Watch." https://www.garmin.com/en-GB/p/665374.

[90] MATLAB-4, "Find local maxima - MATLAB findpeaks - MathWorks United Kingdom." https://uk.mathworks.com/help/signal/ref/findpeaks.html.

[91] MATLAB-2, "Fourier Transforms - MATLAB & Simulink - MathWorks United Kingdom." https://uk.mathworks.com/help/matlab/math/fourier-transforms.html.

[92] P. Heckbert, "Fourier Transforms and the Fast Fourier Transform (FFT) Algorithm," tech. rep., 1995.

[93] MATLAB-3, "Fast Fourier transform - MATLAB fft - MathWorks United Kingdom." https://uk.mathworks.com/help/matlab/ref/fft.html.

[94] D. Lyon, "The discrete fourier transform, part 4: Spectral leakage," 2009.

[95] "Wavelet Analysis of Physiologic Signals - MATLAB & Simulink Example - MathWorks United Kingdom." https://uk.mathworks.com/help/wavelet/ug/wavelet-analysis-of-physiologic-signals.html.

[96] Mathworks, "Wavelet analysis of physiologic signals." https://uk.mathworks.com/help/wavelet/ug/wavelet-analysis-of-physiologic-signals.html.

[97] K. Bawa and P. Sabherwal, "R-peak detection by modified pan-tompkins algorithm," *International Journal of Advancements in Research Technology*, vol. 3, 2014.

[98] A. L. Samuel, "A. l. samuel ibm j. res. develop. vol. 44 no. 1/2 january/march 2000," vol. 44, no. 1, pp. 261–269, 2000.

[99] A. Geron, *Hands-on machine learning with scikit-learn, keras, and TensorFlow*. Sebastopol, CA: O'Reilly Media, 2 ed., Oct. 2019.

[100] A. Varma, A. Sarma, S. Doshi, and R. Nair, "House Price Prediction Using Machine Learning and Neural Networks," *Proceedings of the International Conference on Inventive Communication and Computational Technologies, ICICCT 2018*, pp. 1936–1939, 2018.

[101] B. L. Sturm, O. Ben-Tal, Ú. Monaghan, N. Collins, D. Herremans, E. Chew, G. Hadjeres, E. Deruty, and F. Pachet, "Machine learning research that matters for music creation: A case study," *Journal of New Music Research*, vol. 48, no. 1, pp. 36–55, 2019.

[102] S. Guido and A. C. Mueller, *Introduction to Machine Learning with Python: A Guide for Data Scientists*. 2016.

[103] T. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[104] "Imbalanced data — machine learning." https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data.

[105] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, "SMOTE: Synthetic Minority Oversampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, 2002.

[106] S. Ruder, "An overview of gradient descent optimization algorithms," pp. 1–14, 2016.

[107] M. A. Lones, "How to avoid machine learning pitfalls: a guide for academic researchers," pp. 1–17, 2021.

[108] M. Banko and E. Brill, "Scaling to Very Very Large Corpora for Natural Language Disambiguation," *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pp. 26–33, 2001.

[109] I. T. Jollife and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, 2016.

[110] A. Bhattacharya, "Curse of Dimensionality," *Fundamentals of Database Indexing and Searching*, pp. 141–148, 2014.

[111] M. Mohammed, M. B. Khan, and E. B. M. Bashie, *Machine learning: Algorithms and applications*. No. July, 2016.

[112] B. Peter, *Practical Statistics for Data Scientists*. O'REILLY, 2017.

[113] "Uci machine learning repository: Iris data set." https://archive.ics.uci.edu/ml/datasets/iris.

[114] "Support vector machines." https://scikit-learn/stable/modules/svm.html.

[115] B. Hanin, "Universal function approximation by deep neural nets with bounded width and ReLU activations," *arXiv*, no. 1, pp. 1–9, 2017.

[116] J. A. Yacim and D. G. B. Boshoff, "Impact of artificial neural networks training algorithms on accurate prediction of property values," *Journal of Real Estate Research*, vol. 40, no. 3, pp. 375–418, 2018.

[117] S. Jadon, "Introduction to different activation functions for deep learning." https://medium.com/@shrutijadon/survey-on-activation-functions-for-deep-learning-9689331ba092, Feb 2022.

[118] MIT, "Explained : Neural networks," pp. 1–5, 2021.

[119] IBM - Bluemix, "IBM Bluemix," *IBM.com*, pp. 1–2, 2016.

[120] H. Wang and B. Raj, "A Survey: Time Travel in Deep Learning Space: An Introduction to Deep Learning Models and How Deep Learning Models Evolved from the Initial Ideas," no. March 2017, 2015.

[121] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?," *Advances in Neural Information Processing Systems*, vol. 2018-December, no. NeurIPS, pp. 2483–2493, 2018.

[122] "Typical cnn - Convolutional neural network - Wikipedia." https://en.wikipedia.org/wiki/Convolutional$_n eural_n etwork/media/File$: $Typical_c nn.png$.

[123] "Understanding how to implement a character-based rnn language model - eli bendersky's website." https://eli.thegreenplace.net/2018/understanding-how-to-implement-a-character-based-rnn-language-model/.

[124] R. Dey and F. M. Salemt, "Gate-variants of Gated Recurrent Unit (GRU) neural networks," *Midwest Symposium on Circuits and Systems*, vol. 2017-August, no. 2, pp. 1597–1600, 2017.

[125] J. Riebesell, "Autoencoder." https://tikz.net/autoencoder/.

[126] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice Using MATLAB®: Third Edition*. No. December 2017, 2008.

[127] D. F. Crouse, "A general solution to optimal fixed-gain (α-β-γ etc.) filters," *IEEE Signal Processing Letters*, vol. 22, no. 7, pp. 901–904, 2015.

[128] R. Kalman and R. Buey, "A new approach to linear filtering and prediction theory.," *Trans. ASME, Journal of Basic Engineering*, vol. 83, no. Series D, pp. 95–108, 1961.

[129] R. Turner, "The 1d Kalman Filter," pp. 1–6.

[130] G. Welch and G. Bishop, "An Introduction to the Kalman Filter,"

[131] Espressif, "ESP32 series datasheet." https://www.espressif.com/sites/default/files/documentation/esp32$_datasheet$

[132] "Software — Arduino." https://www.arduino.cc/en/software.

[133] Nxp, "I2C bus," no. October, pp. 1–62, 2021.

[134] "SPI Interface," vol. 1, no. 5, pp. 1–5, 2005.

[135] J. Pickering, "Analogue to Digital and Digital to Analogue Converters (ADCs and DACs): A review update," *CERN Accelerator School: Power Converters, CAS 2014 - Proceedings*, vol. 003, no. May 2014, pp. 363–377, 2018.

[136] Analog Devices, "Single-lead, heart rate monitor front end." https://www.analog.com/media/en/technical-documentation/data-sheets/AD8232.pdf/, 2018.

[137] NXP, "MMA8451Q, 3-axis, 14-bit/8-bit digital accelerometer." https://www.nxp.com/docs/en/data-sheet/MMA8451Q.pdf/, 2013.

[138] P. A. Products, "10K thermistor," no. 151.

[139] "Conductive Rubber Cord Stretch Sensor." https://coolcomponents.co.uk/products/conductive-rubber-cord-stretch-sensor-extras.

[140] H. Kobayashi, "Effect of measurement duration on accuracy of pulse-counting," *Ergonomics*, vol. 56, no. 12, pp. 1940–1944, 2013.

[141] "1555JF42GY Hammond, Plastic Enclosure, Watertight, Wall Mount, Farnell." https://uk.farnell.com/hammond/1555jf42gy/box-flanged-abs-ip66/dp/1829806?ost=1829806.

[142] "Van Damme TourCat Cat5e Flexible Screened Male to Male RJ45 to RJ45 3m - Vdc — CPC UK." https://cpc.farnell.com/vdc/104-212-003/rj45-stage-studio-lead-3m/dp/AV21631?ost=104-212-003.

[143] "RS PRO RJ45 Female Coupler, 1 Port, Shielded — RS." https://uk.rs-online.com/web/p/ethernet-couplers/1116746/?relevancy-data=7365617263685F636173636164655F6F726465723D3126736561726385F696E746572666163655F6E6

[144] I. Cappelli, A. Fort, M. Mugnaini, S. Parrino, and A. Pozzebon, "Underwater to above water LoRa transmission: Technical issues and preliminary tests," *24th IMEKO TC4 International Symposium and 22nd International Workshop on ADC and DAC Modelling and Testing*, pp. 96–101, 2020.

[145] "Ieee standards association." https://standards.ieee.org.

[146] C. Bisdikian, "An overview of the Bluetooth wireless technology," *IEEE Communications Magazine*, vol. 39, no. 12, pp. 86–94, 2001.

[147] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A study of Lora: Long range low power networks for the internet of things," *Sensors (Switzerland)*, vol. 16, no. 9, pp. 1–18, 2016.

[148] "What's the difference between bluetooth and wi-fi?." https://www.britannica.com/story/whats-the-difference-between-bluetooth-and-wi-fi.

[149] "What is wi-fi?." https://3g.co.uk/guides/what-is-wi-fi-explained-in-simple-terms.

[150] N. W. Bergmann, J. Juergens, L. Hou, Y. Wang, and J. Trevathan, "Wireless underwater power and data transfer," *Proceedings - Conference on Local Computer Networks, LCN*, pp. 104–107, 2013.

[151] M. B. Kraichman, "Handbook of Electromagnetic Propagation in Conducting Media," *Naval Surface Weapons Center*, 1976.

[152] Z. Zheng, Y. Fu, K. Liu, R. Xiao, X. Wang, and H. Shi, "Three-stage vertical distribution of seawater conductivity," *Scientific Reports*, vol. 8, no. 1, pp. 1–10, 2018.

[153] S. Ranvier, "Path loss models," *S-72.333 Physical layer methods in wireless communication systems*, no. November, 2004.

[154] S. Farahani, "ZigBee and IEEE 802.15.4 Protocol Layers," *n ZigBee Wireless Networks and Transceivers*, 2008.

[155] "Understanding rssi levels." https://www.metageek.com/training/resources/understanding-rssi/.

[156] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–15, 2015.

[157] D. Ray, T. Collins, S. Woolley, and P. Ponnapalli, "A Review of Wearable Multi-wavelength Photoplethysmography," *IEEE Reviews in Biomedical Engineering*, vol. 15, no. Figure 2, pp. 1–17, 2021.

[158] "Module: tf.keras, TensorFlow Core v2.9.1." https://www.tensorflow.org/api$_d ocs/python/tf/keras$.

[159] A. R. Pérez-Riera, R. Barbosa-Barros, R. Daminello-Raimundo, and L. C. de Abreu, "Main artifacts in electrocardiography," *Annals of Noninvasive Electrocardiology*, vol. 23, no. 2, pp. 2–9, 2018.

[160] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," *2018 International Interdisciplinary PhD Workshop, IIPhDW 2018*, no. May, pp. 117–122, 2018.

[161] "Peak analysis." https://uk.mathworks.com/help/signal/ug/peak-analysis.html.

[162] D. Jaswal, S. V, and K. Soman, "Image Classification Using Convolutional Neural Networks," *International Journal of Scientific and Engineering Research*, vol. 5, no. 6, pp. 1661–1668, 2014.

[163] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. 2016.

[164] Fitbit, "How do I track my sleep with my Fitbit device?." https://help.fitbit.com/articles/en$_U S/Help_a rticle/1314.htm/$, 2020.

[165] H. R. Colten and B. M. Altevogt, *Sleep disorders and sleep deprivation: An unmet public health problem*. 2006.

[166] T. R. Meir Kryger and W. C., *Principles and Practice of Sleep Medicine*. 2017.

[167] J. Malik, Y. L. Lo, and H. T. Wu, "Sleep-wake classification via quantifying heart rate variability by convolutional neural network," *Physiological Measurement*, vol. 39, no. 8, 2018.

[168] "Haaglanden medisch centrum sleep staging database v1.1."