Universität Heidelberg
Hochschule Heilbronn

Master Thesis

Real-Time Body Temperature and Heart Rate Monitoring System for Classification of Physiological Response Patterns Using Wearable Sensor and Machine Learning Technology

Author:
Reto Wettstein
195190

Supervisor:
Dr. Jocelyn Dunstan
Prof. Dr. Steffen Härtel

Examiner:
Prof. Dr. Christian Fegeler
Prof. Dr. Wendelin Schramm

A thesis submitted in fulfilment of the requirements for the degree of Master of Science (M.Sc.)

performed at the

Grupo de Aprendizaje de Máquinas en Biomedicina y Salud (GAMBYS)
Centro de Informática Médica y Telemedicina (CIMT)
Facultad de Medicina
Universidad de Chile

December 17th, 2018

Declaration of Authorship

I, Reto WETTSTEIN, declare that this thesis titled *Real-Time Body Temperature and Heart Rate Monitoring System for Classification of Physiological Response Patterns Using Wearable Sensor and Machine Learning Technology* and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this university.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this university or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

Acknowledgements

First of all, I would like to thank the German Academic Exchange Service (DAAD) for providing the scholarship which allowed me to travel to Chile.

Also, I want to express my sincere gratitude to Prof. Dr. Christian Fegeler, who arranged the contacts to the Universidad de Chile and made my travels possible. Furthermore, my thanks go to the MOLIT Institute for providing the hardware needed to perform this thesis.

Additionally, I would like to thank Prof. Dr. Steffen Härtel, who gave me the opportunity to conduct my thesis at the Universidad de Chile. It was a great pleasure for me to be part of the Centro de Informática Médica y Telemedicina (CIMT). Special thanks go to the members of the Grupo de Aprendizaje de Máquinas en Biomedicina y Salud (GAMBYS) and my office colleagues for all their valuable inputs and their patience with my questions, professionally and also in social matters.

At this point, I would like to express the deepest appreciation to my supervisor Jocelyn Dunstan for her assistance and support during my thesis. Not only did her critical analysis of my progress give me valuable inputs which improved my work, it also made me think about novel approaches to try. She supported me in every possible way without which this thesis would not be what it is today.

Last but not least, I would like to thank my family and friends for listening to all the oral pleadings about my work and the inherent problems I encountered. They gave me valuable advice. Without their help and support this work would not have been possible.

Abstract

Real-Time Body Temperature and Heart Rate Monitoring System for Classification of Physiological Response Patterns Using Wearable Sensor and Machine Learning Technology

by Reto Wettstein

Introduction: A subset of mobile devices allows recording of health data. This facilitates continuous monitoring and analysis of personal vital signs and opens the potential for automated anomaly detection outside of clinical environments. An unobtrusive day-to-day system could be used to detect anomalies caused by disease onset or to improve quality of life for chronically ill patients. But, interpretation of sensor health data is challenging. Physiological response patterns (PRP) depend on demographic factors and are based on activities, diseases and the environmental context. Therefore, PRPs can not be described in general terms. However, machine learning algorithms (MLA) could be used to classify individual PRPs as either normal or abnormal.

Objectives: The objectives of this thesis were to develop an architectural concept of a real-time monitoring and classification system based on the vital signs heart rate and body temperature and to implement it as a proof-of-concept prototype. The prototype should then be used to asses if MLAs can classify a tendency change of vital sign response patterns as either physiologically normal or abnormal.

Methodology: The architectural concept and the prototype were developed using a Cosinuss° One sensor, an Android application and a Python server. The selected MLAs for anomaly detection are Local Outlier Factor, Isolation Forest, One-Class Support Vector Machine and Autoencoder. A 72 hour-long data sample was recorded for assessment of these algorithms (N = 1). Since generation of irregular health data is not possible at the push of a button, the measurements during the activities sport, metro and eating were regarded as artificial anomalies. All the remaining measurements were considered as normal. The MLAs were finally evaluated using confusion matrices to calculate the metrics accuracy, sensitivity and specificity.

Results: The MLAs performed in all cases with an accuracy higher than 80 %. With the exception of the Isolation Forest, specificity was higher than sensitivity. All algorithms showed a specificity higher than 88 %. For sensitivity the MLAs reached results better than 76 %. The best overall results were achieved using the One-Class Support Vector Machine with 89.94 % accuracy, 87.80 % sensitivity and 92.07 % specificity.

Conclusion: This thesis introduces an approach for automated classification of PRPs based on mobile sensor data. The prototype shows that different selected activities can be classified as irregular using MLAs. Further research is needed to asses if also irregularities caused by diseases can be detected with high accuracy, sensitivity and specificity.

Contents

List of Figures

List of Tables

List of Listings

Abbreviations

ANN	**A**rtificial **N**eural **N**etwork
API	**A**pplication **P**rogramming **I**nterface
BLE	**B**luetooth **L**ow **E**nergy
bpm	**b**eats **p**er **m**inute
BT	**B**ody **T**emperature
CNN	**C**onvolutional **N**eural **N**etwork
CSV	**C**omma-**S**eperated **V**alues
FHIR	**F**ast **H**ealthcare **I**nteroperable **R**esources
GATT	**G**eneric **ATT**ribute
HL7	**H**ealth **L**evel **7**
HR	**H**eart **R**ate
HTTP	**H**yper**t**ext **T**ransfer **P**rotocol
ICU	**I**ntensive **C**are **U**nit
IDE	**I**ntegrated **D**evelopment **E**nvironment
JSON	**J**ava**S**cript **O**bject **N**otation
LOF	**L**ocal **O**utlier **F**actor
ML	**M**achine **L**earning
MLA	**M**achine **L**earning **A**lgorithm
OC SVM	**O**ne-**C**lass **S**upport **V**ector **M**achine
ORM	**O**bject-**R**elational **M**apping
OS	**O**perating **S**ystem
PRP	**P**hysiological **R**esponse **P**attern
REST	**RE**presentational **S**tate **T**ransfer
SDK	**S**oftware **D**evelopment **K**it
STU	**S**tandard for **T**rial **U**se
XML	**E**xtensible **M**arkup **L**anguage

Chapter 1

Introduction

Mobile devices are becoming more and more an integral part of our lives [1] and are due to steadily decreasing costs accessible for everyone [2]. According to CCS Insight [3], the sale of wearable devices will increase in the next five years annually around 20 %, therefore reaching a revenue of 29 billion US-Dollars by sells of more than 243 million units in 2022. A subset of these devices (*e.g.* wearables like fitness trackers and smart watches) have a series of sensors that allow recording of health data [4]. This facilitates continuous monitoring and analysis of personal vital signs [5].

1.1 Motivation

Analysis and interpretation of data recorded by wearable devices is challenging [6]. Physiological response patterns (PRP) corresponding to different activities, diseases or a changing environmental context vary from person to person, depending on demographic factors like age, gender, fitness level and other characteristics [7–10] and thus cannot be described in general terms. Additionally, bad performance in analysis and interpretation of this data can trigger unnecessary alarms and therefore lead to alarm fatigue of physicians and nurses [11]. Moreover, early warning scores in non-critical clinical environments are normally calculated three times a day thereby not caching early deterioration of a patient's health [12]. Further challenges arise due to the difficulty of accessing and collecting vital signs associated to diseases and their impact on deterioration of health [6]. But, this data is needed to adjust, train and improve early warning systems.

Machine learning algorithms (MLA) open up the possibility to utilize the collected data in personalized medicine [13], where an algorithm learns to interpret a person's individual PRPs. This interpretation can focus on the recognition of changes and trends in PRPs

and therefore eventually on signs of deterioration of an individual's health status. As a consequence, the combination of MLAs and continuous health data recording allows for real-time classification systems of individual PRPs according to corresponding activities, diseases and the environmental context. The system could act as an early warning system in the case of detected abnormalities.

However, the accuracy, sensitivity and specificity of this approach depends on two main indicators. First, on the quality of the measurements generated by the sensor. If the sensor does not adequately reflect a person's vital signs, the system will not be able to detect abnormalities and thus trigger many false alarms or miss cases of health detoriation. And secondly, on the performance of the applied MLAs. As mentioned above, collecting health data associated to diseases is difficult and therefore training data will consist only of normal vital sings. For this reason only unsupervised and semi-supervised classification algorithms can be considered for such systems. In general, they have worse classification accuracies than supervised approaches [14].

1.2 Objectives

This thesis will focus on an early warning system as introduced in Section 1.1. The goal is to develop a real-time classification system for health anomaly detection using vital sign data produced by an in-ear wearable sensor and MLAs that learn continuously due to a real-time user feedback loop. In order to achieve this goal, various objectives were conceptualized, designed and executed. These objectives are:

(i) Evaluate the measurement quality of the chosen sensor and therefore its suitability for real-time detection of changes in vital signs.

(ii) Develop an architectural concept of a distributed real-time monitoring and classification system.

(iii) Implement this architectural concept as a proof-of-concept prototype.

(iv) Show that MLAs can be used to learn the individual PRPs of a person's vital signs taking into account activities, demographic factors and the environmental context.

(v) Show that MLAs can classify a tendency change of vital signs in response patterns as physiologically normal or abnormal.

The architecture of the distributed real-time monitoring and classification system ((ii) and (iii)) will consist of the following three major components:

(a) A wearable sensor for vital sign measurements (*i.e.* Cosinuss° One sensor measuring body temperature and heart rate).

(b) A server component for storage of the measured vital signs and training of the different MLAs (*i.e.* Python Django Server).

(c) A mobile application as a bridge between the sensor and the server component which sends the measured training data to the server, applies the trained MLAs to new measurements and displays their results (*i.e.* Android application).

These objectives have been selected in such a way that the implementation of this work will make it possible to fully asses the following hypothesis:

> *A real-time body temperature and heart rate monitoring system enables personalized classifications of physiological response patterns as either normal or abnormal.*

1.3 Thesis Structure

The remainder of this thesis is structured as follows: In Chapter 2 the theoretical background of vital signs, their measurement and analysis using machine learning (ML) methods in clinical and non-clinical environments, the implementation of distributed systems and ML for anomaly detection is established. After that, Chapter 3 walks the reader through the applied methodology and the materials, libraries and tools used to develop and test the various components of the proof-of-concept prototype. Chapter 4 presents the architectural concept of the proposed system, its implementation in the proof-of-concept prototype and the differences between these two. Then, in Chapter 5 the results of the Cosinuss° One sensor evaluation and the performance of the selected MLAs are presented. Afterwards, these results are discussed together with the architectural concept and its implementation in Chapter 6 in order to assess the hypothesis presented in this chapter. Finally, this thesis is completed by an extensive outlook about future research topics in Chapter 7.

Chapter 2

Theoretical Background

This Chapter first introduces in Section 2.1 the basics of vital signs, especially heart rate and body temperature, and their continuous monitoring in clinical and non-clinical environments to create an understanding of the application domain and to develop awareness with regard to the problem described in Chapter 1. Furthermore, previous heart rate measurement evaluations of the chosen Cosinuss° One sensor are presented. In addition, the development of distributed systems based on Bluetooth and Representational State Transfer architectures using medical communication standards is discussed in Section 2.2. After that, Sections 2.3 and 2.4 will introduce machine learning for anomaly detection in order to explain the concepts of the technology with respect to the realization of the proof-of-concept prototype.

2.1 Vital Signs

In hospitals, documentation and analysis of vital signs is crucial in order to detect the deterioration of a patient's health state as quickly as possible and subsequently to provide effective treatment. Vital signs that are traditionally measured are body temperature, heart rate, blood pressure, respiratory rate and blood oxygen saturation. [15] In recent years it has also become popular in the general population to monitor personal health using wearable devices. Among other things, these devices can serve as indicators to help change one's lifestyle, optimize workouts, prevent dangers or optimize sleeping behavior. [16]

The vital signs body temperature and heart rate are the ones measured by the selected sensor and are therefore of importance for this thesis. They are discussed in more detail in the following two sections.

2.1.1 Body Temperature

A person's body temperature (BT) is maintained using thermoregulation of hypothalamus functions by producing heat in the body's tissues or releasing heat into the environment through the skin. A healthy person's core BT lays in the range of 36.0 °C to 37.5 °C. Within this range are fluctuations which are dependent on various factors. [17] For one thing, the circadian rythm gives BT a 24h period. The highest BT is measured in the late afternoon or early evening and the lowest in the early morning before waking up. The approximate difference can be around 1 °C depending on various factors of BT variability. This can be seen in Figure 2.1. Also, BT can vary during a womens menstrual cycle and can therefore be used for fertility planning. If the BT is measured shortly after waking up, an increase of 0.25-0.5 °C can be measured around the time of ovulation. Moreover, increasing BT could be measured during high physical activity. People with a general higher physical schedule have a lower BT limit in the circadian rythm. This means that the difference between the daily minimum and maximum is greater than for people who are generally less active. Additionally, in contrast to younger comparative patients, the daily BT amplitude decreases with age. Last, studies suggest a variability of BT depending on the time of the year. However, these results are inconsistent and could have been influenced by other factors. [18]

Variations in BT measurements can be dependent on the measurement site and method. The chosen site and method also influences convenience and reliability of measurements. [18] It can be distinguished between invasive and non-invasive measurement methods. Non-invasive methods include oral, axillia, tympanic membrane and body surface measurements. Invasive methods include rectal (gold standard), oesophagus, pulmonary artery and urinary bladder measurements. Measurements of thermometers are thereby based on thermistors, thermocouple sensors or infrared waves. [19]

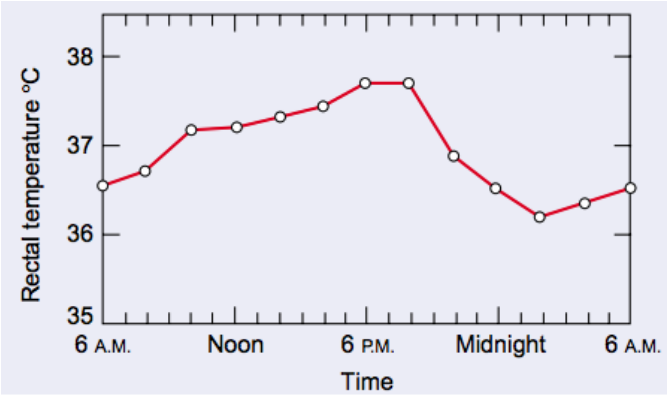


FIGURE 2.1: Influence of the circadian rythm on body temperature, following [17].

Exposure to high or low environmental temperatures can have an influence on BT and lead to hyper- or hypothermia. Both cause untreated disabilities and can lead to death. Hypothermia is the event where the BT drops below the normal range. Other reasons for hypothermia include infections and bacteremia, ethanol or drug ingestion, a central nervous system event, cachexia from malignancy or malnutrition, gastrointestinal bleeding and endocrine deficiencies. A BT exceeding the normal range can be hyperthermia or a fever. Hypothermia, sometimes as well called heat stroke, is induced by the body's inability to loose enough heat in comparison to an uncontrolled heat production. It can be a common occurrence during prolonged stays in very hot and humid regions. Other possible reasons for hyperthermia include neuroleptic malignant syndrome, serotonin syndrome, endocrinopathy, cerebral hemorrhage, hypothalamic injury and drug-induced hyperthermia. [20] Hyperthermia should not be confused with fever, which is a controlled increase of BT induced by the hypothalamus thermoregulation. Reasons for fever include viral and bacterial infections, microbial toxins, mediators of inflammation or immune reactions. [21]

2.1.2 Heart Rate

Heart rate (HR) measures the number of contractions of the heart as beats per minute (bpm) and is regulated by the synchronous cooperation of the sympathetic and parasympathetic nervous system [22]. A healthy person has a resting HR between 60 and 100 bpm. Athletic people tend to have a lower resting HR than 60 bpm, dropping sometimes as low as 40 bpm. HR is adjusted depending on the physical activity and therefore on the body's need of oxygen and the removal of carbon dioxide. Various factors can change the HR based on the resting frequency. [23] The most important ones are according to the American Health Association [23]:

- **Air temperature:** When temperature or humidity rises, the heart pumps a little more blood so that the HR increases by 5 - 10 bpm.
- **Body position:** Regardless of the position (lying, sitting or standing), the HR is usually the same, but can increase for a short time when the position is changed.
- **Emotions:** Emotions like stress, anxiety, happiness, sadness or pleasant and unpleasant surprises can lead to elevated HR.
- **Body size:** Obese people can have a higher resting HR.
- **Medication use:** Depending on the medication, HR can either increase (*e.g.* beta blockers) or decrease (*e.g.* thyroid medication).

HR can be measured either manually or using electronical devices. Manual measurements are done by feeling and counting an artery's pulsation. Best spots include the radial artery at the wrist and the carotid artery at the neck. However, manual HR measurements can be done at all body parts where an artery can be felt. [24] Measurements using electronical devices are often based on electrocardiography by recording the electrical activity of the heart and deriving the HR from its pattern. HR can also be calculated by pulse oximeters, which use the optical method of photoplethysmography. Light of the green or infrared spectrum is emitted into the blood and the reflected values are measured. Due to the pulsation of the blood, the measurement of the reflected values is also pulsating and can be used to derive the corresponding HR. [25] One special kind of photoplethysmography is the circummission-method which is used in the selected sensor. The method is documented in the dissertation of Rieger [26].

Changes in HR can also be triggered by diseases. A resting HR of constantly above 100 bpm is called tachycardia. Causes include damage to heart tissue from heart diseases, congenital abnormality of the heart, anemia, smoking, fever, alcohol or drug abuse. [27] If the HR is constantly beating less then the normal 60 - 100 bpm, we talk about bradycardia. It only represents a problem if not enough oxygen is distributed by the heart. Causes include heart tissue damage related to aging, damage to heart tissues from heart disease or heart attacks, myocarditis, imbalance of chemicals in the blood, such as potassium or calcium, or obstructive sleep apnea. [28] Both, tachycardia and bradycardia, can lead to heart failure, sudden cardiac arrest or death, tachycardia additionally to stroke.

2.1.3 Vital Signs Monitoring in Clinical Environments

The five vital signs mentioned in Section 2.1 are generally continuously monitored with other parameters in clinical environments like the intensive care unit (ICU). In the most common systems, deterioration of a patient's health is detected if measured values exceed upper or lower thresholds of value intervals regarded as normal. Additionally, data driven and knowledge based rules are applied to detect irregularities. [29] Often more than 40 sources of alarms can be identified. The strength of an alarm then depends on the magnitude of the threshold violation, the time and duration of the violation, and the number of vital parameters that trigger the alarm. This results in an alarm hierarchy. In many cases there are no default settings for thresholds, so they may vary depending on the monitoring system. [30] Systems have to be constructed following the international standards IEC 60601-1-11 and IEC 80001. One of the biggest problems with current available systems is the rate of false alarms, leading to alarm fatigue. To avoid missing valid alarms, a large number of false alarms have to be accepted. Different studies show that 72 % to 90 % of all alarms in a hospital environment are triggered unnecessarily,

causing medical personnel to be exposed to an excessive number of alarms, which in turn can lead to desensitization and missed alarms. [11]

Literature shows many approaches based on machine learning to reduce the number of false alarms. Imhof et al. [31] provided a detailed summary of statistical approaches and alarm algorithms which can be used for ICU patient monitoring. Ben Rejab et al. [32] proposed a monitoring system based on Support Vector Machines reducing false alarms. Chen et al. [33] used a Random Forest based approach to increase alarm accuracy. Zang [34] applied artificial neural networks to generate patient individual clinical alarms, achieving a sensitivity of 96 % and a specificity of 99 % by using eight hours of training data.

In comparison to high risk units, vital signs are monitored in general wards mostly three times a day and interpreted using a scoring system. However, there is no continuous monitoring and therefore the deterioration of a patients health is detected with delay, increasing the mortality rate. [12] For this reason, Weenk et al. [12] evaluated the feasability of continuous monitoring of vital parameters with portable devices in the general ward. They applied the ViSi Mobile and HealthPatch system and showed promising results regarding the consistency of the measurements and the acceptance of the systems by the patients. Cardona-Morrell et al. [35] reviewed multiple approaches for continuous monitoring of vital signs in general wards and did not find any improved patient outcomes. Kamio et al. [36] showed in a review of twelve studies that compared to traditional methods, MLAs can improve prediction of clinical deterioration. Salem et al. [37] propose a framework for anomaly detection in medical wireless body area networks. They show that their approach maintains a higher true positive and lower false negative rate compared to similar frameworks.

2.1.4 Vital Signs Monitoring Outside Clinical Environments

Tracking of vital signs and fitness values becomes more and more popular in the general population. It gives people the opportunity to track and improve their own health. Even health insurers have now started to pay out rewards and benefits when fitness trackers and other wearables have been able to prove the achievement of certain goals. [38] This is also reffered by the term patient empowerment. Fitness trackers and wearables allow recording of many different parameters ranging from the number of steps, covered distance, calories burnt, sleep time or sleep quality to HR and BT. A good overview of many available devices ordered according to their application range can be found in [39] or [40].

However, it only makes sense to use these recordings to improve personal health if the measurements of the various parameters are accurate. Since the Cosinuss° One sensors measures BT and HR, the following evaluations focus on these measurement types. El-Amrawy and Nounou [41] compared multiple wearable devices able to measure HR with a Onyx Vantage 9590 professional clinical pulse oximeter while performing a certain number of steps on a treadmill. They found an accuracy of over 90 % for all tested devices. In [42], Gilinov et al. tested various HR wearables at rest, light, moderate and vigorous intensity. The accuracy of the tested HR monitors varied based on the type of training. It is highest on the treadmill and lowest on the elliptical trainer. Ge et al. [43] compared accuracy of chest and wrist warn HR monitors (electrocardiography respectively photoplethysmography based measurements). While in normal conditions the measurement types showed almost identical results, recordings during exercises could deviate as much as 10 %. Overall, chest warn monitors appeared to be more accurate.

There are not many wearable devices and fitness trackers with the availability to measure BT. To the best of the author's knowledge, there has been no publication until the date of this thesis' publication about their accuracy.

2.1.4.1 Cosinuss° One Heart Rate Evaluation

Evaluation of the Cosinuss° One sensor HR measurement quality has been done by many sport professionals. Schlichenmaier [44] compared its HR measurements with those of a Garmin HR chestbelt during a 40 min endurance run. The superimposed similar measurement curves can be found in Figure 2.2. The recorded graph of the Cosinuss° One sensor is more erratic because it consists of more measurement points and thus the course of HR is presented more realistic. In contrast, the curve measured by the Garmin HR chestbelt is smoothed. The mean HR measured by both devices is similar: the Cosinuss° One sensor returned a HR mean of 154 bpm and the Garmin chestbelt 155 bpm. Similar findings are presented in [45] and [46], in the later during cycling.

To the best of the author's knowledge, a comparison between Cosinuss° One BT measurements and other measurement techniques to quantify its measurement quality has not been published yet. Therefore, this is conducted as part of this thesis (see Section 5.1).

More about the Cosinuss° One sensor including its specification can be found in Section 3.2.1.

For real-time analysis and interpretation of vital signs measured by the Cosinuss° One sensor or other wearable devices, a system able of collecting, storing and analyzing the data is needed. This thesis proposes a distributed system for this purpose. The theoretical background of such a system type is introduced in the following section.

FIGURE 2.2: Comparison of heart rate measurements between a Garmin heart rate chestbelt (red) and the Cosinuss° One sensor (blue), following [44].

2.2 Distributed Systems

A distributed system is a system that consists of multiple components. Each individual component is autonomous and located on different networked computers. They communicate with each other via messages so that actions can be coordinated to achieve a common purpose. [47] The user of the system perceives the different components as a single unit. In contrast, an individual component only has a limited incomplete view of the system required to perform its subtask. [48]

There are two approaches to distributed systems: centralized and decentralized. A centralized approach is mostly organized using a client-server paradigm. The server provides centralized services and functionalities to multiple clients simultaniously. A client can consume these services by sending a request and then waiting on the servers reply. Therefore, the communication between a server and its clients is asymmetric. Centralized client-server architectures can be built as multiple tiers where one server simultaneously provides a service for clients and accesses another server as a client itself. A decentralized approach is mostly established as peer-to-peer architecture. Participants in such a system are equal. Communication between peers is symmetric and therefore peers can act at the same time as server and client. In addition, there are hybrid distributed system architectures which consist of centralized and decentralized parts. [49]

Connections for message exchange between the individual components depend on the architectural concept and can be established in various ways. Since Bluetooth Low Energy based Generic Attributes Profiles and web based Representational State Transfer

are used for messaging in the developed proof-of-concept prototype, they are discussed in the following sections.

2.2.1 Bluetooth Low Energy and Generic Attributes Profiles

Bluetooth is an industry standard (IEEE 802.15.1) for point-to-point communication between devices at short distances using short-wavelength UHF radio waves in the ISM band of 2.4 GHz. The full specification of Bluetooth can be found in [50].

As an extension of the Bluetooth standard, Bluetooth Low Energy (BLE) was published for applications in the areas of mobile phones, wearables and the Internet of Things. Compared to standard Bluetooth, BLE has been designed in such a way that significantly cheaper and more energy-efficient connections are possible within the same reception range. [51]

Message exchange between connected BLE devices is established using Generic Attributes (GATT) Profiles. A profile is a hierarchical data structure, its composition can be seen in Figure 2.3. Each profile consists of a use-case description, roles and general behaviors of the GATT functionality. Profiles are composed of several services that are needed to fulfill its use-case. To each service belong multiple characteristics. A characteristic consists of a universally unique identifier based type, a value, properties about supported operations and security permissions. It can also contain descriptor meta data and configuration flags. With this framework it is possible to exchange all type of messages, including discovery and connection establishement of BLE devices as well as reading, writing and notification of a characteristics changed value. [52]

2.2.2 Representational State Transfer

Representational State Transfer (REST) is an architectural style for creating web services to ensure interoperability over the Internet and was introduced by Fielding [53]. Its core elements are resources and representations. A resource is any storable type of information (*e.g.* a document, an image, a collection of resources). They consist at least of the information to be stored, an identifier and meta data (*e.g.* a source link or alternates). Actions on resources are performed by REST components using representations. Representations (*e.g.* a HTML document or a JPEG image) show the current or intended state of a resource and are used to transfer them between REST components. They are composed of their data and describing meta data (*e.g.* the media type or the last-modified time).

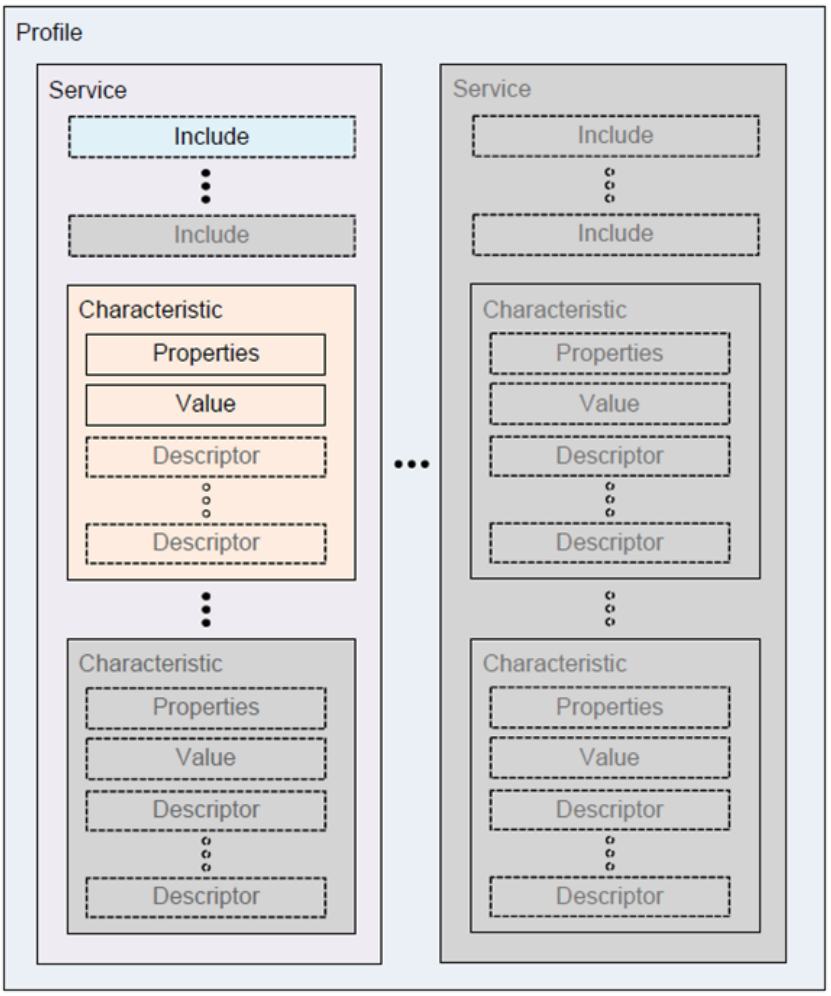


FIGURE 2.3: Hierarchical data structure of a Bluetooth Low Energy Generic Attributes profile, following [52].

A REST architecture consist of the following constraints, which define the properties of a web service [53]:

- **Client–server architecture:** A component's only concern should be its own tasks. This supports portability and scalability and ensures that components can evolve individually.
- **Stateless:** The server stores no context information of the client. The session state is stored on the client. As a consequence, each request to the server has to contain all necessary information so that the server can fulfill the requested task. This ensures visibility, reliability, and scalability of web service architectures.
- **Cache:** Responses to requests have to be defined as cacheable or not-cacheable to ensure that clients do not use outdated or inappropriate data for further requests. This reduces unnecessary traffic and improves efficiency, scalability and user-perceived performance. But, it can also reduce reliability.

- **Uniform interface:** This is one of the most important constraints, since it separates REST from other web service architectures. By applying the software design principle of generality and therefore decoupling implementation and service, components can evolve independently. This simplifies the architecture and increases visibility of interactions. In the same time it decreases efficiency. An uniform REST interface itself has four constraints:
  - Resources in requests can be identified individually.
  - Holding a representation of a resource including its meta data is enough information for manipulation of the resource.
  - Messages are self-descriptive. They hold all the information needed for their processing.
  - Hypermedia should be used as the engine of the application state. This means that a client does not need a hard-coded representation of the REST service. It dynamically receives, based on former requests, all at the time available and potentially need hyperlinks.
- **Layered system:** In a multi-layer system, a component can only see the components with which it is communicating and not any components beyond. This reduces the system's complexity and promotes its independency. As a downside it increases latency. Additionally, by using intermediate layers, components can be split and therefore simplified. Moreover, the scalability of the system can be increased and thus legacy services and clients encapsulated.
- **Code-on-demand:** Transfer and execution of scripts and applets can increase a clients functionality for a short time. In other words, increasing a clients extensibility leads to reduced client complexity and thus increases interoperability between clients and web services. This is the only optional constraint because it can decrease visibility.

A web service application programming interface (API) following the REST architecture is called a RESTful API. It consists of a base uniform resource locator, telling where the web service can be reached, a media type that defines in which format a resource is transferred (*e.g.* application/json, application/xml or text/html) and a standard Hypertext Transfer Protocol (HTTP) method that makes a claim about the nature of the request (*e.g.* creation, modification or deletion). [54]

However, a REST architecture with a RESTful API does not make any assumption on how exchangeable data is structured. Communication standards are responsible for this task. One of these standards for electronic health data is Health Level 7 Fast Healthcare Interoperable Resources.

2.2.3 Fast Healthcare Interoperable Resources

Fast Healthcare Interoperable Resources (FHIR) is a standard developed by Health Level 7 (HL7), describing data formats and elements for electronical exchange of healthcare information. HL7 is a known standardization organization and has already published several health care interoperability standards such as HL7 v2 , HL7 v3 containing a reference information model and the clinical document architecture. Since digitization does not stop at medicine either, methods for a structured and standardized exchange of health data supporting discoverability, availability and understandability of personal electronic health records are needed. The basic component in HL7 FHIR is a resource, describing all exchangeable content. The main characteristic of a resource are a common way to define them using a set of data types, a set of meta data and a human readable part. In addition, HL7 FHIR follows the 80/20 % approach for its resources. This means that a data field is only present when 80 % of the implementations will use it. The remaining 20 % can be added individually through so-called extensions. As format for data representation JavaScript Object Notation (JSON) or Extensible Markup Language (XML) can be used. HL7 FHIR is not yet a normative standard, rather it is classified as a standard for trial use (STU). [55]

HL7 FHIR is used in this project as the data format for message exchange between the mobile phone application and the server component to transfer vital signs measured by the wearable sensor. Once the data has been exchanged, it has to be analyzed and interpreted. ML, which is presented in the following sections, is one possible way of doing this.

2.3 Machine Learning

Machine learning (ML), a part of artificial intelligence, is a research field that is concerned with how computers can acquire the ability to learn. MLAs learn by deriving a general description of the task to be solved from a large amount of examples. These examples are not only memorized by the algorithms, but they also try to derive general laws so that unknown data can be assessed. In general, more difficult problems can be solved if more training data is available. In comparison to manually programming algorithms, this type of learning is often more efficient and cost effective. [56] ML can be divided in four major categories [57]:

- **Supervised learning:** The examples used by an algorithm consist of the actual input values and an additional label indicating the correct output of the examples.

The algorithm is in search of a function which correctly maps the inputs to the corresponding outputs.

- **Unsupervised learning:** No labels are known in advance. The algorithms try to detect hidden patterns in the provided input data.
- **Semi-supervised learning:** A mixture between the two above, combing labeled and unlabeled data for training.
- **Reinforcement learning:** An algorithm learns a strategy depending on the expected behavior. Each of his actions lead to a reward or a punishment. The algorithm then tries to maximize the received reward.

Another important differentiation point between MLAs is the form of their output. If the learned function of a MLA produces a discrete value, we talk about classification. If the output is continuous, it is called regression. [58]

MLAs often have many parameters which can be tuned to receive the best classification results. When tuning a MLA, not only the choice of the values of the individual parameters is important, but also their interaction. A method used to find the optimal parameters is called grid search. This is a search through a manually defined subset of the parameter space, where the global optimum of parameter values and their combinations is found by trying all possible pairs. [59]

Among the two biggest problems in training of MLAs are overfitting and the curse of dimensionality. The term overfitting is used if a classifier can classify the training data with high accuracy, but fails with similar unknown data. A generalized representation of the training data could therefore not be derived. Curse of dimensionality means that many algorithms work well in low dimensions but have problems finding a generalized representation on high dimensional input data. The more dimensions the input data consists of, the further apart are the samples and therefore more samples are needed for accurate training. [56]

When dealing with time-series data, *i.e.* data that was recorded over a period of time where the sequence plays an important role, there are according to Susto et al. [60] two main approaches. One possibility is a data driven approach, where time-series are compared directly on the basis of their raw data. Using this method, each individual data point of a time-series represents one input dimension. Another statistically based approach is to calculate features from time-series, which are then used to compare the data. An advantage of the direct comparison is that no time-consuming feature extraction has to be calculated. Likewise, no information from the original signal is lost. The second method has advantages especially if time-series do not have the same length and if they

have a variable number of missing data points. Whether one of the two methods or a mixed version can be used often depends on the quality and length of time-series.

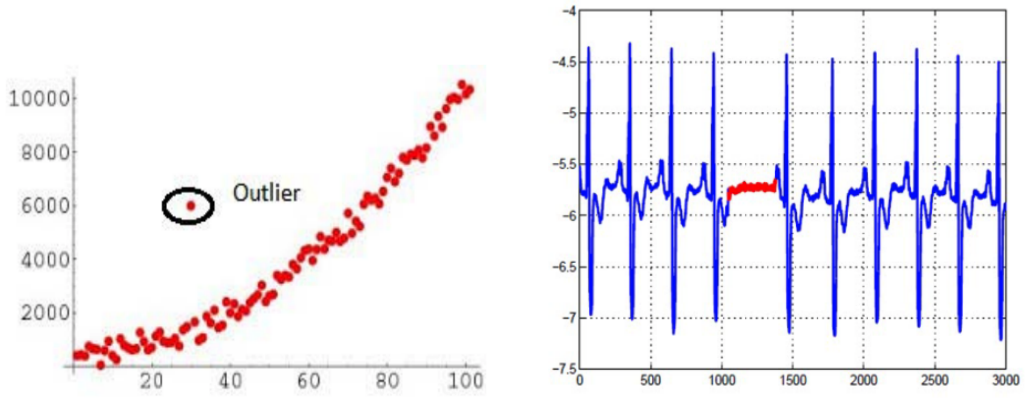
There are many applications of ML in many different fields. Of interest for this thesis are unsupervised and semi-supervised classification algorithms developed for anomaly detection, because the distribution between normal and abnormal vital sign time-series of a subject is not uniform. In addition, abnormal data cannot be generated at the push of a button and is therefore usually not available for training of the algorithms.

2.4 Anomaly Detection

An anomaly is according to Hawkins [61] "*an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism*". Therefore the detection of anomalies (synonimously called outlier detection, novelty detection, deviation detection or exception mining [62]) in data science is the identification of objects, events or observations that do not match expected patterns or other objects in the data set [63].

There are many use-cases in which anomaly detection can be useful. Examples include fraud detection (*e.g.* in credit card transactions based on purchase behavior), sport statistics (*e.g.* in various parameters which explain a players' performance gain or loose), measurement errors (*e.g.* in different types of sensors) or personal health and medicine (*e.g.* unusual vital signs, symptoms or test results can be indicators of health deterioration) [64].

An important aspect in anomaly detection is the nature of an anomaly. They are classified on the basis of their occurrence. Generally, three types of anomalies are distinguished, namely point anomalies, contextual anomalies and collective anomalies. Point anomalies are the simplest type (see Figure 2.4a). They occur when a data point is very different from all other data points. An example would be one BT measurement of 38 °C in a time-series of BT measurements between 36.5 °C and 37 °C. A contextual anomaly is one that is considered abnormal due to the conditions of its environment. A heart rate of 180 bpm can be normal during high intensity training and abnormal in a desk working environment. If there exists a relationship between data points in a data set and one relationship is different from all other relationships, then we talk about a collective anomaly (see Figure 2.4b). An example of a collective anomaly is a change in the RR-interval by a certain factor between two heart beats. [65] In this thesis all three kinds of anomalies play an important role for anomaly detection in vital signs as explained by the examples mentioned before.



(a) (b)

FIGURE 2.4: Outlier examples: (a) point anomaly, (b) collective anomaly, following [65].

Identification of anomalies is a non-trivial task. The following general factors from Chandola et al. [66] have the biggest influence on successful anomaly detection in vital signs:

- The characterization of all normal values of a type of data recording is not always straightforward, especially for values at the threshold between normal and abnormal data.
- The size of the delta, which decides whether a value change is regarded as an anomaly, depends on the domain. In medicine, small changes can have already a big impact on the patients health status, whereas small fluctuations in stock markets are normal.
- Normal values and anomalies can change over time. Vital signs particularly depend on a patients current state like age or fitness level.
- Recorded data is often noisy, making it difficult to distinguish between actual anomalies and contaminated measurements. This is especially the case when the difference between normal and abnormal values is small.
- One of the biggest challenges is the presence of accurate training and validation data for fitting anomaly detection algorithms.

As the list shows, different types of anomalies have different degrees of detection severity. The further down the list, the more difficult it is to solve the problem described.

Additionally, choosing the right algorithm that achieves the best results is difficult. It depends on multiple factors including the size, quality and distribution of the data. In many cases it cannot be said which algorithm will perform best before trying multiple ones. [67] But, as principle guidance, there exist algorithm cheat sheets developed for

example by Microsoft [67], SAS [68] or scikit-learn [69]. They are based on different heuristics regarding the type of available data and the problem to be solved.

As already mentioned in the introduction chapter, two other challenges arise in medical anomaly detection. First, vital signs, symptoms or test results of medical conditions can be very different for each individual person. They can be influenced by the patient's environment or demographic factors and hence cannot be generalized. Therefore, algorithms used for health anomaly detection have to be applied on data collected for each person individually, taking into account their demographic factors, their normal activities and their environmental context in order to classify PRPs correctly. Secondly, alarm fatigue is a major issue regarding anomaly detection in health data. Therefore, the sensitivity and specificity of anomaly detection algorithms has to be maximized to reduce the false positive and false negative alarm rates respectively.

There exists three types of scenarios for detection of anomalies similar to the ML categories mentioned above. They influence the selection of possible algorithms [64]:

- **Supervised scenario:** Training sets with normal and abnormal data are available. However, the distribution of classes is very unbalanced, mostly only a few instances of abnormal data objects are present.
- **Semi-supervised scenario:** The training set consists of only one of the two classes. Either data objects of the normal or the abnormal class are present.
- **Unsupervised scenario:** In many applications, there is no training data available for which it is known whether the data is normal or abnormal.

The literature presents many methods and algorithms to identify anomalies in all of the three scenarios. According to Kriegel et al. [64] they can be divided into two main categories:

Statistical approaches:

- Methods based on statistical tests
- Depth based algorithms
- Deviation based methods

Model based approaches:

- Distance based algorithms
- Density based algorithms

Explanation of all possible algorithms would be out of scope for this thesis. For a good overview the reader is therefore redirected to [64], [70] or [71]. The algorithms selected in this thesis to identify anomalies are Local Outlier Factor, Isolation Forest, One-Class Support Vector Machine and Autoencoders. They are introduced in the following sections.

2.4.1 Local Outlier Factor

Local Outlier Factor (LOF) is a density based approach introduced by Breuning et al. [72]. The idea of this approach is to compare the local density of a point with the density of the point's k nearest neighbors. The density calculation is based on determining the maximum radius of a circle around an evaluated point needed to enclose the k nearest neighbors. A point with a high density belongs to a cluster, a point with a low density is considered an anomaly. Figure 2.5 illustrates this approach by evaluating if point p belongs to cluster C. $MinPts = 3$ indicates the k nearest neighbors which are considered for the density estimation. Since the density of p is a lot lower than the density of its neighbors, the probability of an anomaly is high. The output of the approach is not a binary decision, but rather a decision based on a factor computed by the quotient between the average density of the closest neighbors and the density of the evaluated point. A quotient close to 1 is considered as normal. With a higher quotient increases the probability of an anomaly.

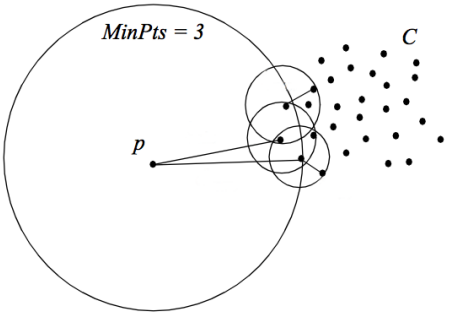
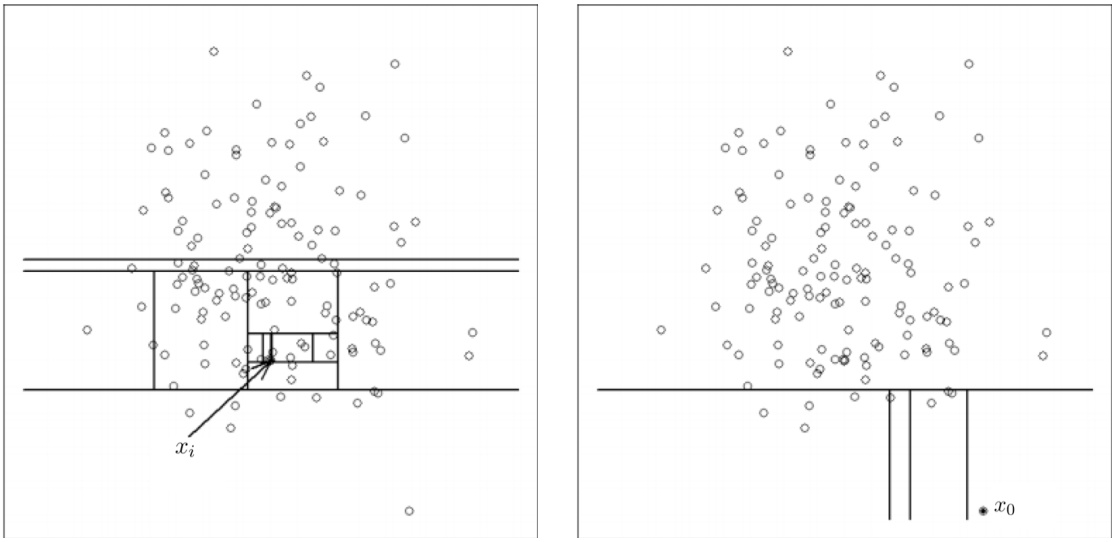


FIGURE 2.5: Basic idea of local density estimation used by the Local Outlier Factor classifier, following [72].

2.4.2 Isolation Forest

An Isolation Forest is a distance based approach introduced by Liu et al. [73]. The method relies on an assembly of Decision Trees. Decision Trees are hierarchically ordered decision rules. They are displayed as a tree diagram to classify objects. [74] The basic idea behind an Isolation Forest is to generate Decision Trees which isolate an instance from the rest of other instances. Because anomalies are rare and different from normal data, the chance of isolation is higher. The generation of such trees is based on recursive data partitioning, selecting randomly a feature and then selecting randomly a split value between the maximum and minimum value of the selected feature, until each instance is isolated. As a consequence, paths to anomalies are way shorter since fewer instances need fewer partitions until isolation and instances with more distinguishable values have

a higher chance to be separated in early stages. This can be seen in Figure 2.6. In (a) a point x_i, identified as normal instance, needs twelve partitioning steps. In contrast, an anomaly point x_0 in (b) needs only four partitioning steps until isolation. Assembling multiple isolation trees to a forest leads to converging path lengths for both normal and abnormal data and therefore to consistent classifications. The algorithm returns an anomaly score between 1 and 0 based on the average path length over all trees. A value close to 1 has a shorter path length than the average path length and thus is an anomaly. Values closer to 0 have a longer path length than the average path length and hence are normal values. A value of 0.5 indicates that the path length of a value is close to the average path length.

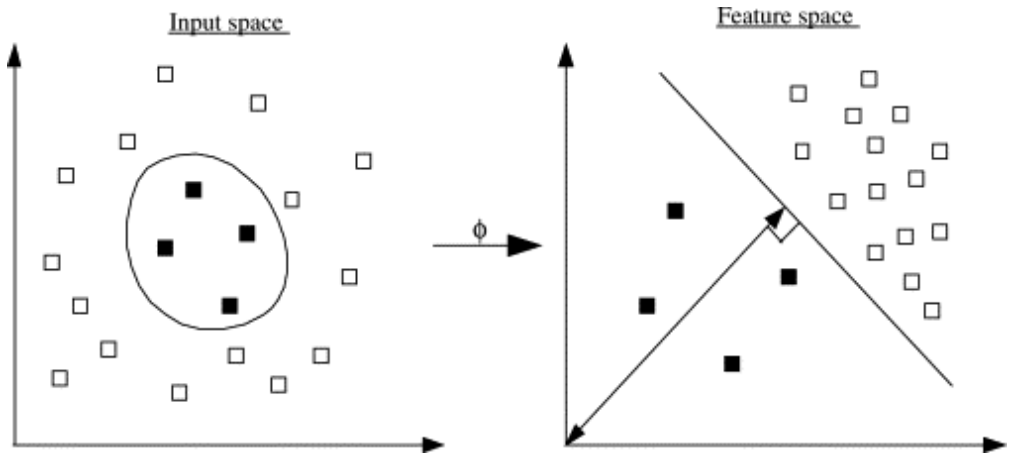


(a) (b)

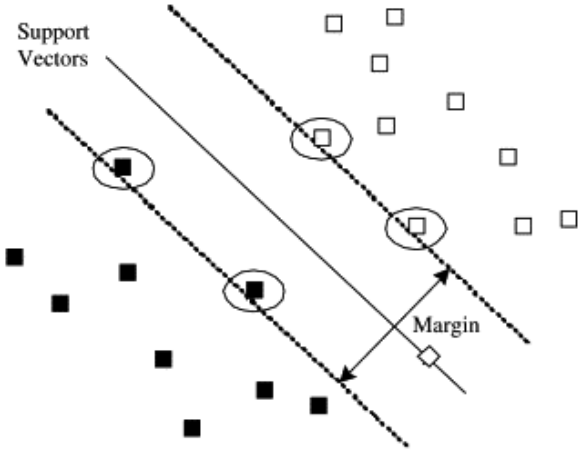
FIGURE 2.6: Partitioning tree examples: (a) a tree of a normal data point x_i, (b) a tree of an abnormal data point x_0, following [73].

2.4.3 One-Class Support Vector Machine

The One-Class Support Vector Machine (OC SVM) was introduced by Scholkopf et al. [75]. The basics of this method is the supervised learning approach of normal SVMs. SVMs are a classifier that try to map nonlinear separable data points from the input space to a higher dimensional feature space using a kernel function. The goal of this transformation is to make the data points linearly separable. The classification is based on the search for a hyperplane which optimally separates different classes in the way that the margin between the hyperplane and support vectors, crossing the data points closest to the hyperplane, is maximized. [76] This can be seen in Figure 2.7.



(a)



(b)

FIGURE 2.7: Support Vector Machine illustration: (a) a kernel function ϕ mapping the input space to a higher dimensional feature space, (b) an optimal hyperplane which maximizes the margin to the support vectors, following [77].

The OC SVM is an unsupervised extension to normal SVMs for anomaly detection. It separates all data points from the origin of the feature space by searching a hyperplane that maximizes the distance between the data points and the origin. An example is shown in Figure 2.8. The resulting binary function will return $+1$ in a small area of the feature space for normal data and -1 for the remaining unknown or anomalous data [78].

2.4.4 Autoencoder

Autoencoders were first proposed by Rumelhart [80] and Baldi and Hornik [81], primarily as a data compression technique. In addtition to compression, they can also be used for anomaly detection. Autoencoders are based on the supervised learning method of artificial neural networks which are computational models inspired by its biological counterparts [82].

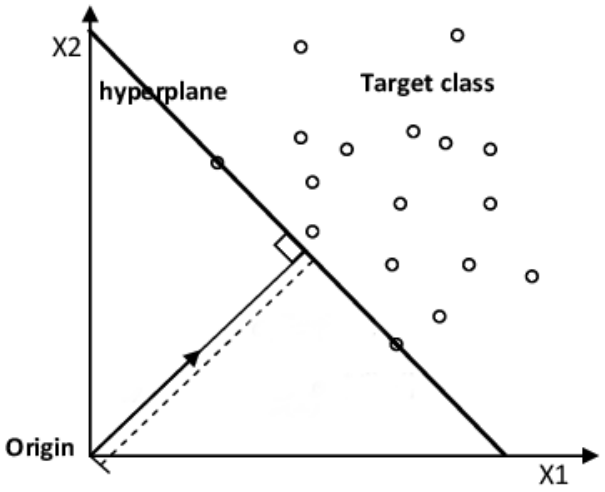


FIGURE 2.8: Illustration of a One-Class Support Vector Machine maximizing the distance between the hyperplane and the origin in the feature space, following [79].

2.4.4.1 Artificial Neural Networks

Artificial neural networks (ANN) have their fundaments in the research of McCulloch and Pitts [83], and Hebb [84]. They introduce the term ANN and make a comparison of there computational model to biologial neurons. The most basic ANN consists of a single neuron as seen in Figure 2.9. The main components are the input vector x and the weights vector w, which is multiplied and summed with the input vector. The transformation function decides wether the neuron is firing or not by generating its output y as either 1 (firing) or 0 (not firing). [82] There are many transformation functions, the ones most commonly used are the step-, linear-, sigmoid-, tanh- and the relu-function [85]. An additional bias term b can be added to the activation function to shift the transformation function. This simple ANN has very limited functionality and can only be used for simple linear classification tasks.

For more complex tasks, feed-forward multi-layered ANNs (see Figure 2.10) are used. They consist of three parts: an input-, an output- and multiple hidden layers. The input layer has the same number of artificial neurons as the training object has dimensions. The output layer consists of the same number of neurons as there can be different classification results. The hidden layers transform the input in a way that a correct classification can be derived. Their number depends on the classification problem to be solved. This setup allows solving of many nonlinear classification problems. [82]

In order for ANNs to fulfill their classification task, they have to learn in a supervised manner by examples. This learning is normally based on the gradient descent or the backpropagation algorithm. These two algorithms adapt in every learning epoch the weights vector w, trying to minimize a cost function by propagating the classification

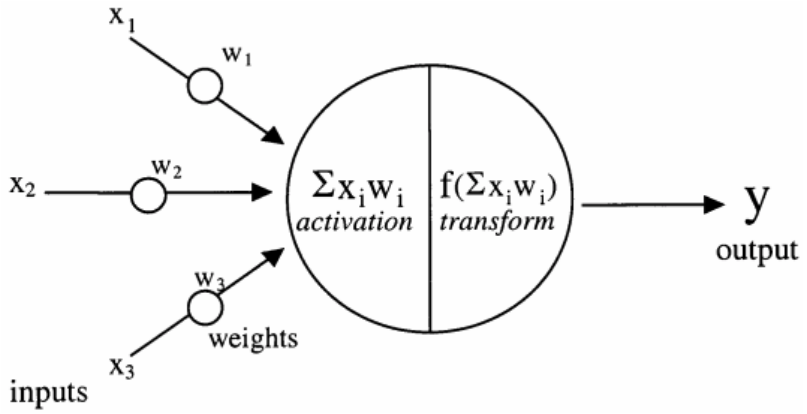


FIGURE 2.9: A single artificial neuron, following [82].

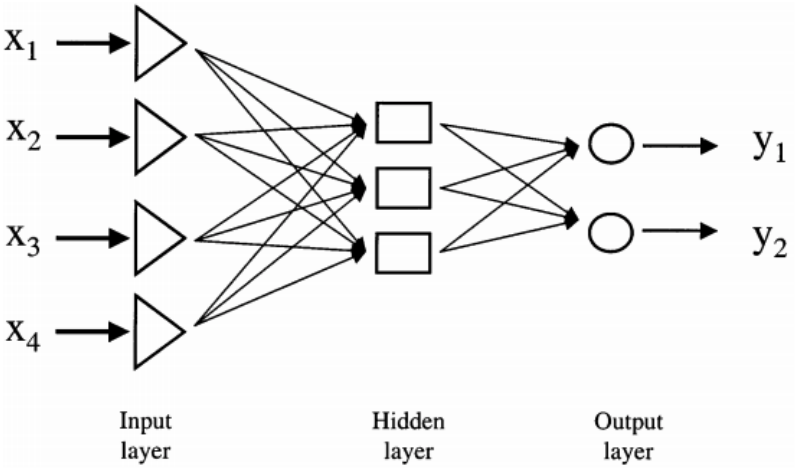


FIGURE 2.10: Architecture of a multilayer feed-forward artificial neural network, following [82].

error back through the network. The cost function calculates the error of a classification made during training and therefore defines the magnitude in which the weights are adjusted. A good overview of the different possible cost functions can be found in [86].

There are many different kinds of multi-layered ANNs, all developed for a specific application field. An overview of the most used types of ANNs is given in [87]. The one most interesting for this thesis are Autoencoders.

2.4.4.2 Autoencoders for Anomaly Detection

In comparison to other feed-forward multilayer ANNs, Autoencoders are an unsupervised or semi-supervised ML approach. The goal of an Autoencoder is to learn the representation of an input as a lossy compressed model, storing only the most important information. A possible architecture can be seen in Figure 2.11. It consists of two major

nonlinear mappings, an encoder and a decoder, with symmetrical design and multiple layers. The encoder's task is to find a function f_Θ, mapping an input vector x to a reduced form z, which is called latent space representation. In other words, z represents the essential characteristics of an input object and thus the encoder performs an analysis of the input data's most important components. In contrast, it is the decoder's task to find a function g_Θ, which maps the reduced form z into the reconstruction vector $\hat{x}$ in the original input space, with as few errors as possible. The difference between the original input and the reconstructed output of the Autoencoder is called the reconstruction error. Because anomalous data is different to the normal data used for training, it can be expected that the reconstruction of anomalous data is more error prone. Therefore, an unseen input is considered as an anomaly if the reconstruction error is greater than the maximum reconstruction error of the training data. [88] By moving the threshold of the maximum reconstruction error, the sensitivity and specificity of an Autoencoder can be adapted in regard to its classification result.

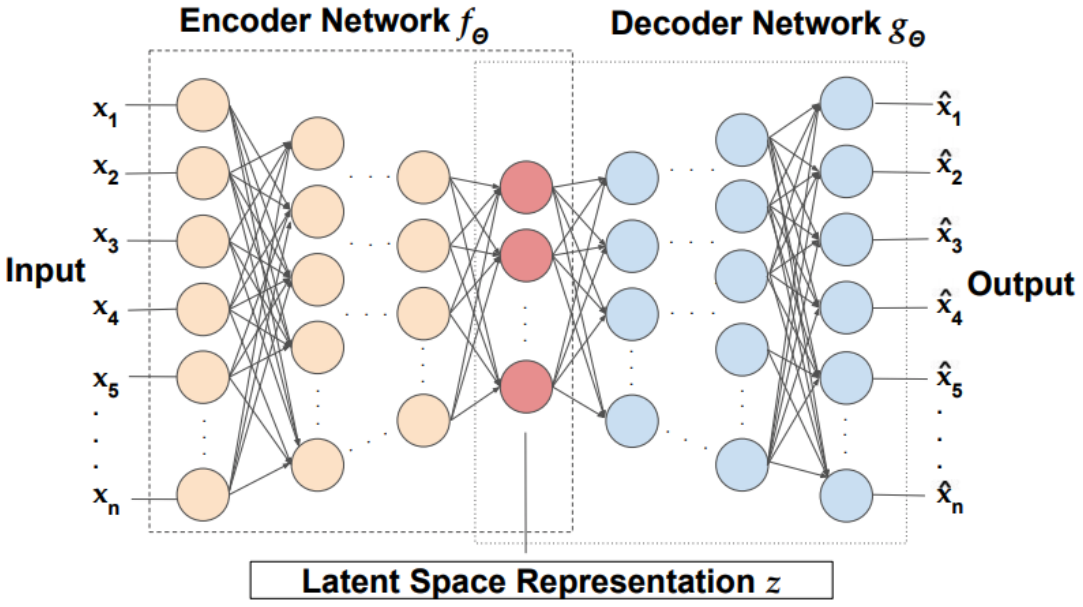


FIGURE 2.11: Architecture of an Autoencoder, following [88].

Chapter 3

Methodology

This chapter describes the selected approach to achieve the goals of this thesis. Section 3.1 lists chronologically the executed tasks to evaluate the Cosinuss° One sensor and to implement the classification system as a proof-of-concept prototype. It explains the decisions taken to fulfill the objectives and to asses the hypothesis. Then, Section 3.2 presents the specification of the Cosinuss° One sensor and the different materials, libraries and tools used during the implementation phase of the proof-of-concept prototype.

3.1 Applied Approach

To asses the hypothesis from the first chapter, the objectives from Section 1.2 were divided into individual work packages to conceptualize, implement and test the proposed real-time classification system. Additional work packages were added to evaluate the measurement quality of the selected Cosinuss° One sensor. Finally, the following chronologically listed approach was chosen and realized:

(i) Development of an architectural concept for a real-time classification system and planning of its realization in a proof-of-concept prototype.

(ii) Selection of a sensor for measurements of different vital signs.

(iii) Selection of a platform and programming language for the mobile phone application and its implementation.

(iv) Selection of a programming language for the ML server and its implementation for data storage.

(v) Recording of measurements for the evaluation of the Cosinuss° One senor.

(vi) Evaluation of the Cosinuss° One sensor to show its suitability for real-time detection of irregularities in BT and HR.

(vii) Selection of MLAs and their implementation as part of the ML server.

(viii) Planning of an experiment to evaluate the prototype including the performance of the selected MLAs.

(ix) Deployment of the implemented proof-of-concept prototype on different hosting platforms for different parts of the system.

(x) Execution of the experiment and evaluation of the recorded results.

The following sections explain these steps in more detail by describing how they were carried out and why decisions were made.

3.1.1 System Development and Implementation

The developed architectural concept (i), its implementation in a proof-of-concept prototype and the differences between the concept and the actual implementation are described in its own chapter. The description can be found in *4. Real-Time Classification System.*

The sensor chosen for measuring vital sings (ii) is called Cosinuss° One. It was selected because of its ability to measure BT and HR in one and the same device, its supposed to be very accurate measurements, its open source API to collect data, its comfortable way of wearing in the ear, its long battery life and its mid-tier price segment. Its specification is listed in Section 3.2.1.

Android was chosen in step (iii) for the implementation of the mobile phone application because it has the biggest market share of all smart phone operating systems (OS) [89]. In addition, implementation of Android applications is simplified with the provided software development kit (SDK) from Google in the Java programming language. Furthermore, the available libraries and tools, especially for the development of RESTful API clients, the amount of tutorials and the active community facilitate implementations of such applications.

To implement the ML server (steps (iv) and (vii)), Python was selected as the preferred programming language for the reason of its low complexity and versatility. With Python it is possible to build easy to understand applications in every domain of computer science. These Python applications stand out for their readability, changeability and maintainability. Especially in ML, the ease of understanding and the application of available supporting programs make fast prototyping possible. Additionally, the provided

stack of open source libraries make it most interesting in data analysis and ML projects. There are repositories for fast low-level mathematical calculations and data processing, but also repositories consisting of high-level input-output mechanisms and ML libraries with preimplemented MLAs.

Communication between the Cosinuss° One sensor and the Android application was predefined by the sensors abilities and thus implemented using BLE. HL7 FHIR was selected for the communication between the Android application and the ML server, since it is an upcoming standard for interoperable medical data exchange. It would enable the Android application to send the measured data not only to the ML server of this thesis, but also to any HL7 FHIR enabled system (*e.g.* an electronic health record). This means that the measured data could be used for any health related task in hospitals or by a general practitioner.

3.1.2 Cosinuss° One Body Temperature Measurement Evaluation

Since evaluations of the Cosinuss° One sensor for HR already exist (see section 2.1.4.1), focused this thesis on the evaluation of its BT measurements. In order to evaluate the sensor's quality, the BT measurements were assessed for general validity and compared with measurements of a reference thermometer (steps (v) and (vi)). Due to available resource constraints, it was not possible to make a comparison with another continuously measuring in-ear thermometer. Therefore, measurement differences and inequalities had to be accepted. However, the tendencies of both thermometers should be the same. Measurements of the Cosinuss° One sensor were taken in five second intervals and labeled in order to be able to track which activity led to which measurements (for available labels see Section 3.1.4). These measurements were then compared to measurements of a commercially available digital thermometer, as it can be bought in any pharmacy. The measurements of the reference thermometer were taken manually in the armpit using a two minute interval.

Comparison of the measurements was done on one subject (N = 1) in different settings. First in a non-active environment to verify the general validity of the Cosinuss° One measurements. Then, during sports on an elliptical trainer to evaluate the impact of activity. Additionally, measurements were analyzed in different surroundings inside and outside of buildings in order to determine the effect of the surrounding temperature on the measurement quality. Due to a lack of resources a bigger sample size of users for the evaluation was not possible. The same applies for the later evaluation of the MLAs.

The Cosinuss° One evaluation results can be found in Section 5.1.

3.1.3 Selection and Implementation of Machine Learning Algorithms

The vital sign data in this thesis is self generated using the sensor and therefore the class of normal vital sign measurements is known. For this reason, the applicable algorithms originate from the field of unsupervised and semi-supervised anomaly detection scenarios. The MLAs were selected in step (vii) according to a number of aspects so that they are based on different concepts and mathematical approaches (for the differences between the selected MLAs see Sections 2.4.1 - 2.4.4). This allows not only the compare the algorithms themselves, but also to compare the algorithm types and therefore to decide which one is best suited to classify PRPs. The MLAs that were chosen are:

- Local Outlier Factor
- Isolation Forest
- One-class Support Vector Machine
- Autoencoder

Because of the selected libraries, the first three algorithms build their architecture and select the threshold for distinguishing between normal and anomalous data during training themselves. In contrast, the ANN architecture of the Autoencoder has to be built manually before training and the threshold computed afterwords. The architecture of the ANN consists of a three level encoder and decoder respectively. Its input and output size were decided by the length of the vital sign time-series to be analyzed. Hidden layers have a size counting 75 % and 50 % of the neurons from the input layer. The full network can be seen in Table 3.1. The reconstruction error of the Autoencoder was used to calculate the threshold differentiating between normal and anomalous data using the metric mean absolute error. The selected threshold is the mean reconstruction error plus 1.5 times the standard deviation of the reconstruction error over all training data.

3.1.4 Prototype Test and Machine Learning Algorithms Evaluation

To be able to test the developed system, it had to be ensured that access to the proof-of-concept prototype was possible from everywhere and not only in the development environment. For this reason, the ML server was deployed (ix) to a Raspberry Pi exposed to the Internet and the Android application installed on a Motorola G5S.

The final experiment to evaluate the system and the chosen MLAs (steps (viii) and (x)) with regard to their ability to classify PRPs correctly as normal or abnormal, was done as a self-experiment (N = 1) using an approach consisting of two phases. In the first

Table 3.1: Autoencoder model summery.

Layer	Type	# Units	# Parameters
Input	InputLayer	56	0
Encoder 1	Dense	42	2394
Encoder 2	Dense	28	1204
Decoder 1	Dense	28	812
Decoder 2	Dense	42	1218
Output	OutputLayer	56	2408
Total parameters:			**8036**

phase for training data collection, BT and HR measurements were recorded continuously for three consecutive days (72 hours), measuring these vital signs in five second intervals, same as during the evaluation of the Cosinuss° One sensor. Measurements were suspended when the sensor needed to be charged. During this data collection phase, the measurements were labeled to be able to trace which activity led to which measurements and to divide measurements into normal and abnormal data for training and test purposes. The labels were chosen based on two categories, activity and location. The activity labels were selected as basic activities on a high level as normal test data (left side) and more special activities which are executed some time during a daily schedule, but not very often, so that they could reflect artificial anomalies (right side). The available activity labels are:

- Sleeping
- Lying
- Sitting
- Walking
- Sport
- Metro
- Eating

The label walking stands for both activities, standing and walking. The labels for the location labeling were selected to additionally see the influence of the surroundings on the sensor's measurements (*e.g.* environment temperature and other influences on the subject) and were only used for the evaluation of the Cosinuss° One sensor. Available labels are:

- Inside
- Outside

In the second phase, the selected MLAs had to be evaluated according to their performance. Since generation of abnormal health data from a healthy person is not feasible or predictable, the recorded normal data had to be split into training and test data. Data labeled as sleeping, lying, sitting and walking, regardless of whether they were measured inside or outside of a building, were considered as normal values and were therefore exclusively used for training of the MLAs. Except the same amount of normal data as there was abnormal data was randomly reserved for the test set and not used during training. Data labeled as eating, metro and sport was considered as abnormal measurements and only used in the test set.

Sport, metro and eating were selected as anomalous data, because they stand out from the normal measurement types in three different degrees of difficulty. Sport should be the easiest to identify. It can be assumed that sport will cause a major change in BT and HR. Therefore, sport was selected as a label to show that the algorithms can identify changes in PRPs. Metro and eating were selected as a bigger challenge for the algorithms, because they should produce very similar measurements to other activity labels. Eating is usually done during sitting, so these measurements should be similar. In the metro a person is normally standing or walking, therefore measurements should be similar to these labels when recorded inside a building. Visualizations for comparison of BT and HR measurements for these labels can be found in Appendix A. In comparison to the normal training data, only a fraction of measurements were taken of the anomaly labels for testing purposes. The initial data collection phase led to 65944 measurements with normal data and 2743 measurements for abnormal data (719 sport, 836 metro and 1170 eating). This results in a ratio of 1:25 for abnormal data and shows why sport, metro and eating could be used as artificial anomalies. The test set of abnormal data was then supplemented by the same number of normal data, previously held out from the training set, to create test classes of equal size in order to not distort the classification result of the MLAs.

After recording, the continuous vital sign measurements were split into individual chunks according to the selected analysis time of two minutes, using an overlap of 30 seconds between two time-series. This period is long enough to recognize short-term changes in vital signs. Shorter time-series do not allow the recognition of significant changes. When longer time-series are used, events that have led to changes in vital signs are less visible because the surrounding readings obscure them. Using an overlap between the generated time-series ensures that significant changes at the border between two consecutive time-series could be recognized too.

Next, the MLAs were trained using a data-driven approach. Grid search was applied for a manually selected set of parameter combinations in order to find the best one for each

classifier. The use of this type of parameter tuning ensures that the best possible result is achieved for each MLA.

Then, performance evaluation of the MLAs was done in an overall setting and for each type of abnormal data individually. A confusion matrix was used to calculate the accuracy, sensitivity and specificity of each MLA. In contrast to normal confusion matrices, where normal data has the label 0 and special data the label 1, it is here the other way around. Normal vital sign time-series have the label $+1$. Anomalies or abnormal vital sign time-series have the label -1. Therefore, a confusion matrix consist of the following cells:

- **True positive:** An abnormal vital sign time-series is correctly identified as an anomaly.
- **True negative:** A normal vital sign time-series is correctly discarded as no anomaly.
- **False positive:** A normal vital sign time-series is incorrectly identified as an anomaly.
- **False negative:** An abnormal vital sign time-series is incorrectly discarded as no anomaly.

An example of a confusion matrix using these cells can be seen in Table 3.2. Considering these definitions, sensitivity measures the percentage of time-series measurements who are correctly identified as being anomalies (*i.e.* the ratio of true positives divided by the sum of true positives and false negatives). Specificity measures the percentage of time-series measurements who are correctly identified as not being anomalies (*i.e.* the ratio of true negatives divided by the sum of true negatives and false positives).

Additionally, in case of the Autoencoder, the reconstruction error was calculated to show the difference between the error of normal and abnormal data. Moreover, it shows the influence of the selected threshold to divide the measurements into normal and abnormal recordings and thus the effect of the threshold on sensitivity and specificity.

The MLA evaluation results can be found in Section 5.2.

3.2 Development Materials, Libraries and Tools

The development of the proof-of-concept prototype and the computation of the MLAs during the training phase was done on an Apple MacBook Pro 2014 running OS X

TABLE 3.2: Example of a confusion matrix for vital sign time-series anomaly detection.

		PREDICTED CLASS	
		−1 / Abnormal Measurements	**+1 / Normal Measurements**
ACTUAL CLASS	**−1 / Abnormal Measurements**	True Positives	False Positives
	+1 / Normal Measurements	False Negatives	True Negatives

High Sierra version 10.13.5 with a 2.6 GHz Intel Core i5 CPU, 8 GB 1600 MHz DDR3 RAM and a 256 GB SSD. For data generation and testing of the above described experiment, the ML server was deployed on a Raspberry Pi Model 3B v1.2 running Linux Debian derivative Raspbian 9 Stretch version 4.9.80-v7+ 32Bit with a 1.2GHz Broadcom BCM2837 CPU, 1GB LPDDR2-900 SDRAM and a 16 GB microSD. The mobile phone as host for the Android application was a Motorola G5S running Android Nougat version 7.1.1 with a 1.4 GHz Snapdragon 430 Octa-core 1.4 GHz Cortex-A53 CPU, 3 GB LP DDR3 RAM and 32 GB internal storage.

3.2.1 Cosinuss° One Sensor

The Cosinuss° One is the sensor of choice for measuring vital signs. According to its website it is "*a professional fitness tracker monitoring multiple vital signs with stunning accuracy*" [90]. Unlike other fitness sensors worn on the wrist or chest, the Cosinuss° One measures the vital signs in-ear. The manufacturer promises that this would be more comfortable due to an ergonomic design, the skin-friendly material and the ease of handling. The sensor should offer its wearer more freedom of movement compared to other fitness trackers.

The sensor has the ability to measure HR, HR variability and BT. These measurements are based on the company's proprietary technology called earconnect. HR and HR variability is measured optically using the circummission-method. BT is determined by means of a resistance sensor. The specification of the BT and the HR sensor can be found in Table 3.3. The vital signs are then calculated with intelligent algorithms, digitized and wirelessly forwarded to a connected device. The size of the sensor is four by four centimeters and it weighs six grams. It is therefore the smallest and lightest HR monitor commercially available. Its battery should last up to eight hours under real-world conditions and recharge in one hour. The manufacturer's recommended price

Table 3.3: Cosinuss° One body temperature and heart rate sensor specification, following [91].

	Property	Value
Heart Rate	Sensor type	Pt1000
	Measurement method	Resistance temperature detector
	Measurement accuracy	±0.1 °C
	Measuring range	0 - 50 °C
Body Temperature	Sensor type	Optical sensor
	Measurement method	Circummission-method
	Measurement accuracy	± 1 beats per minute (bpm)
	LED spectrum	Green

for the Cosinuss° One sensor is 119 €. The full specification of the sensor can be found in [91].

The Cosinuss° One sensor can be used together with all devices supporting connections via BLE and Ant+. The measurements can be displayed using the companion application available for iOS and Android. Unfortunately the HR variability is only measured so far but not implemented on the companion apps. Due to the fact that the Cosinuss° One sensor uses open source Bluetooth GATT profiles as messaging standard, it is also possible to write own applications for reading measured data.

3.2.2 Android Development

The development of the Android application was done using Java version 1.8.0_161 and Android SDK version 26. The preferred integrated development environment (IDE) was Android Studio version 3. Other main libraries used include HAPI-FHIR and Retrofit.

HAPI-FHIR [92] is a library providing an open source implementation of the HL7 FHIR specification in Java. It allows among other things the processes of creating, editing and transforming HL7 FHIR resources between different formats on client devices such as Android applications. The library version used in this thesis was 3.2.0 for HL7 FHIR version Standard for Trial Use 3 (STU3).

The library Retrofit [93] is an open source type-safe HTTP client for Java, especially for Android application development, which makes it comparatively convenient to exchange

structured data such as JSON formats with a web service providing a RESTful API. The version used was 2.4.0.

3.2.3 Server Development

To develop the ML Server, the Python programming language version 3.6.4 was used. PyCharm in the Community Version 2017.3 was the preferred IDE. Other applied libraries include Django and its extension Django-REST-Framework, Smart-on-FHIR Python client, scikit-learn, Keras and Tensorflow as well as Numpy, Pandas and Matplotlib. All of these libraries are open source.

3.2.3.1 RESTful API

To develop the REST architecture of the ML server, the library Django [94] version 2.0.5 and its extension Django-REST-Framework [95] version 3.8.2 were used. Django is a high-level Python web framework for fast, secure and scalable development of web services with a clean and pragmatic design. Adding the extension Django-REST-Framework makes it even more easy to build RESTful APIs based on Django.

The Smart-on-FHIR Python client [96] is another open source implementation of the HL7 FHIR specification developed for the Python programming language. It can be applied for client queries to HL7 FHIR servers and to process FHIR resources, similar to the abilities of the HAPI FHIR library for Android. The library version used in this thesis is 3.2.0 for HL7 FHIR version STU3.

3.2.3.2 Machine Learning

For the development of the server's ML part, several libraries have been exploited. Functionality from Numpy [97] version 1.14.3 and Pandas [98] version 0.22.0 was applied for pre- and postprocessing of the time-series data (see Section 4.2.2). Numpy is one of the most essential libraries for scientific computing in Python providing low-level functionality based on multidimensional arrays and functions to perform fast operations and calculations on these arrays. Pandas is very similar to Numpy in the way that it also provides high performance computing data structures and tools for data analysis. Compared to Numpy, Pandas however provides more high-level functionality, which is especially advantageous when processing time-series and labeled data from relational databases.

All the selected MLAs, except the Autoencoder, were developed using the scikit-learn [99] library version 0.19.1. This library is one of the most popular for ML in Python. It provides easy to use functionality for data mining and data analysis. The functionality ranges from classification, regression, clustering, dimensionality reduction and model selection to pre- and postprocessing of data and evaluation of algorithms. The provided MLAs are already preimplemented and therefore easy to use by setting an algorithm's parameters and providing the input. Evaluation of the algorithms is done by applying preimplemented performance metrics.

For the developement of the Autoencoder, the high-level library Keras [100] version 2.2.0 was used. It provides a flexible API for fast development and experimentation of all kinds of neural networks. It runs on a more low-level neural network library, also called backend in the Keras language, facilitating the functionality provided by the backend. Keras can run on top of many different backends. The one chosen for the proof-of-concept prototype implementation is Tensorflow [101] version 1.5.0. Tensorflow is a library for high-performance computations on dataflow graphs for different tasks and often used for neural network implementations. For the development of the Autoencoder on the MacBook Pro, the standard 64 Bit library of Tensorflow could be used. For deployment of the proof-of-concept prototype on the Raspberry Pi, the library had to be recompiled into a 32 Bit version, since the Raspbian OS is 32 Bit based and a 32 Bit Tensorflow version is not provided yet by its development team.

The generation of many figures presented in the results and discussion chapters was done utilizing the library Matplotlib [102] version 2.2.2. Matplotlib is a Python plotting tool that allows to create high quality figures (*e.g.* plots, bar charts, histograms, scatterplots or errorcharts).

Chapter 4

Real-Time Classification System

This chapter describes first in Section 4.1 an architectural concept for a real-time classification system of vital sign time-series data. Section 4.2 is then concerned with the realized proof-of-concept prototype and its differences to the architectural concept. It presents the implementation of the different components including an Android application and a Python Django server, and discusses the used communication protocols.

4.1 Architectural Concept

The overall architectural concept of the real-time classification system is designed as a distributed system using a multitier client-server architecture. It consists of a wearable sensor, a mobile application and a ML server. A visualization of this architecture can be seen in Figure 4.1. The wearable sensor is used for continuous measurement of different vital signs. The recorded data is then sent to the mobile application using a wireless communication protocol, if supported. The mobile application (client) is responsible for the initiation of the connection to the sensor (server) and the subscription to the communication service. In turn, the sensor will send the requested vital sign measurements either in a predefined interval or as part of a notification service if the measured vital sign values change.

In addition, the mobile application is used to display the received vital sign measurements from the sensor and to relay them to the ML server for training purposes. The ML server is in charge of many different mobile applications, which represent the clients. Its responsibility is to store the received training data and to carry out the training of all available MLAs. The training should be done in regular intervals to make sure that new received training data (since the last training phase) can be incorporated as well. Using

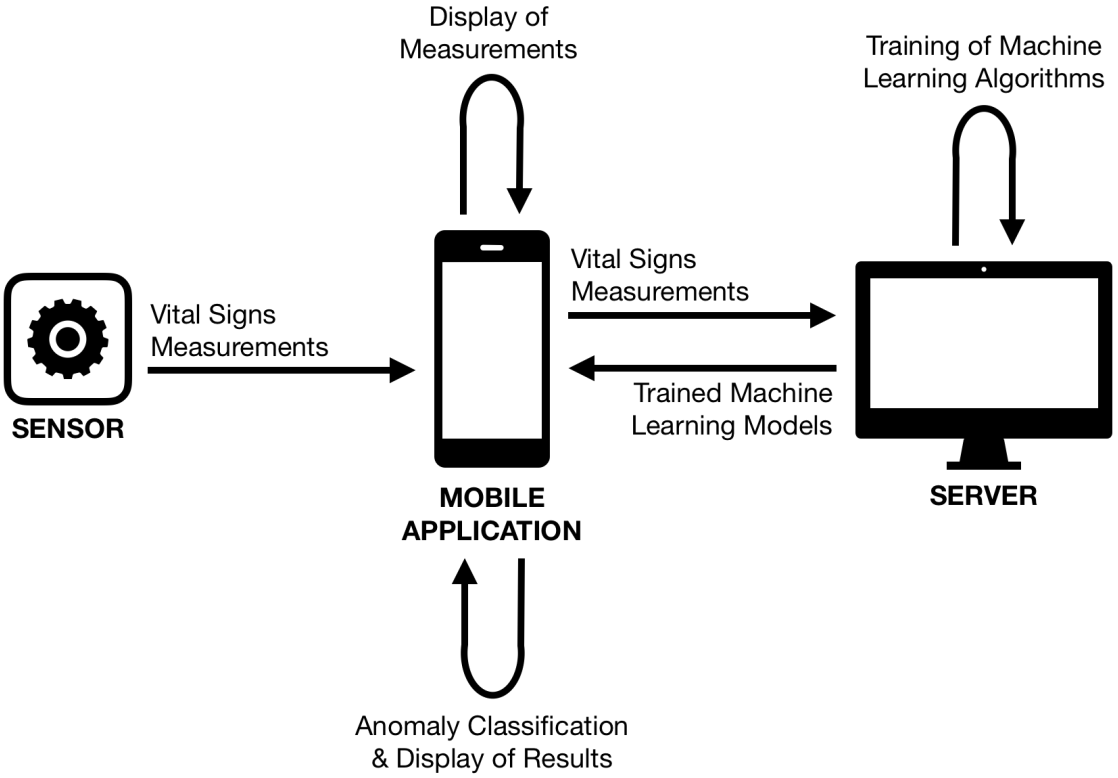


FIGURE 4.1: Overall architectural concept of a real-time vital sign classification system.

this approach, continuous learning and therefore continuous improvement of a classifiers robustness can be satisfied. This is done on the server side since saving the training data and conducting the training needs a lot of computational resources, especially if grid search is used for parameter tuning of an algorithm.

Storage of the training data and training of the MLAs is done in a person-related fashion, resulting in trained models for each individual subject using the real-time classification system. The mobile application can pull the trained model of each algorithm from the server after training. Afterwards, new received sensor data, which is unknown to the trained classifiers, can be applied to the trained models in order to detect possible anomalies. This can be done directly on the mobile device since computational resources for prediction are way smaller compared to the ones needed during the training process. An advantage of executing the classification by the MLAs directly on the mobile device is that the dependency on an Internet connection is no longer given. This eliminates one of the biggest sources for errors, since, after the collection of the training data, the system can be used in places without present Internet connection like in elevators or remote regions.

Moreover, if a measurement should be classified as an anomaly by one of the MLAs, the user needs to have the ability to provide feedback by confirming whether the anomaly classification was correct or false positive. In the case of an incorrect classification, *i.e.*

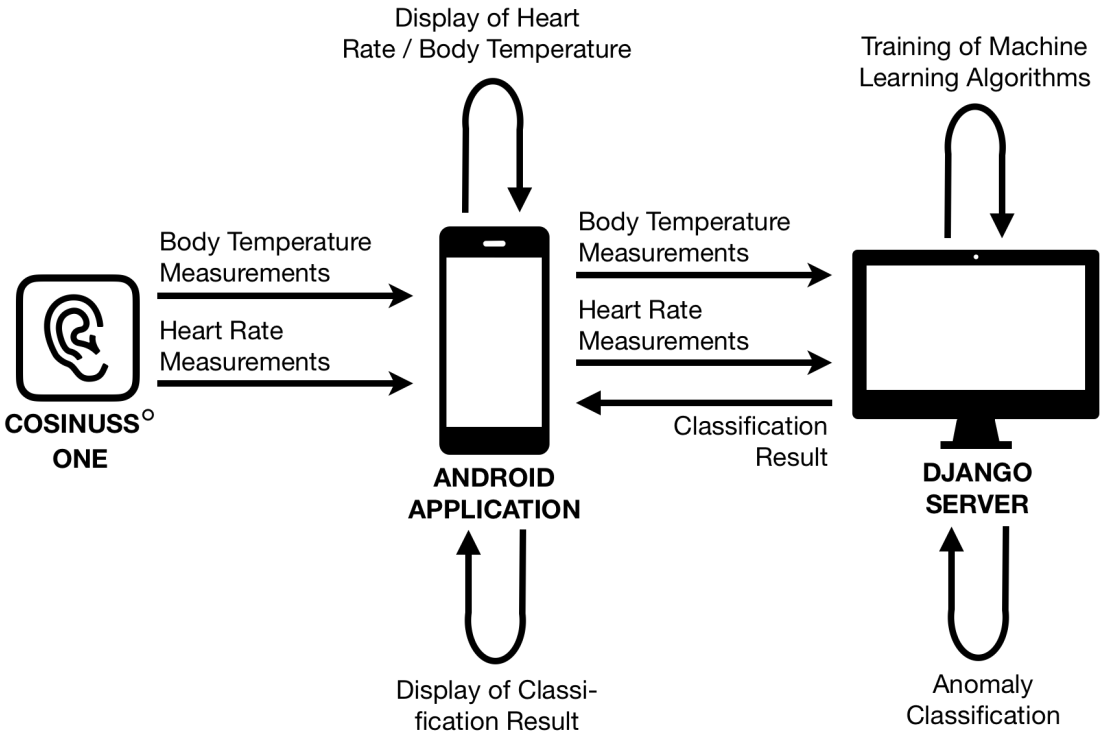


FIGURE 4.2: System architecture implemented in the proof-of-concept prototype.

the time-series is not an anomaly, the measurements that have led to the false positive evaluation of the algorithm are sent to the server as new training data. As a result, new measurements, which are similar to the ones wrongly classified as anomaly, will in the future no longer be classified as false positive. If no Internet connection is available during a false positive classification, the new training data should be stored temporarily until a new connection can be established.

This architectural concept has as consequence that the system has to be used in two phases. First, the user has to generate training data (72 hours are suggested to generate a sufficient baseline of a daily routine's normal measurements, if a five second interval is selected, see Section 3.1.4) before the user can use it for actual classifications of their vital signs.

4.2 Proof-of-Concept Prototype

For simplicity reasons, the implementation of the proof-of-concept prototype differs in some aspects from the intended architectural concept introduced in the previous section. The changes can be seen in Figure 4.2.

The first difference between the figures is that the different components of the system are now accurately specified. The sensor is the Cosinuss° One, measuring the vital signs

BT and HR, the mobile application is defined as an Android application displaying the measurements of the sensor and the ML server is a Python Django server used for the ML tasks.

Another difference is that the execution of the MLAs for classification of the vital sign time-series is done on the server side, like the training of the algorithms, and not as intended directly on the mobile application. The rational behind this decision is that porting the trained algorithms from the server to the client side during implementation would have added another complexity layer to this thesis and therefore would have taken too much time. That is why the mobile application in the proof-of-concept prototype sends the measured data for classification purposes to the server and requests a classification in the same interval. The server then classifies the last received measurements and returns the classification result to the client, where it will be displayed.

The last difference is concerned with the user feedback. It was not implemented because of time reasons and because it has no influence on the performance evaluation of the algorithms. As a result, false positive classifications for the same measurements will also occur for future measurements in the prototype.

The client-server communication between the Android application and the Cosinuss° One sensor is based on BLE GATT services and characteristics. The ones used are:

- The Health Thermometer Service[1] storing the data to be transmitted in the Temperature Measurement Characteristic[2].
- The Heart Rate Service[3] storing the data to be transmitted in the Heart Rate Measurement Characteristic[4] .
- The Battery Service[5] storing the data to be transmitted in the Battery Level Characteristic[6].

[1] `https://www.bluetooth.com/specifications/gatt/viewer?attributeXmlFile=org.bluetooth.service.health_thermometer.xml`, accessed on 09.07.2018

[2] `https://www.bluetooth.com/specifications/gatt/viewer?attributeXmlFile=org.bluetooth.characteristic.temperature_measurement.xml`, accessed on 09.07.2018

[3] `https://www.bluetooth.com/specifications/gatt/viewer?attributeXmlFile=org.bluetooth.service.heart_rate.xml`, accessed on 09.07.2018

[4] `https://www.bluetooth.com/specifications/gatt/viewer?attributeXmlFile=org.bluetooth.characteristic.heart_rate_measurement.xml`, accessed on 09.07.2018

[5] `https://www.bluetooth.com/specifications/gatt/viewer?attributeXmlFile=org.bluetooth.service.battery_service.xml`, accessed on 09.07.2018

[6] `https://www.bluetooth.com/specifications/gatt/viewer?attributeXmlFile=org.bluetooth.characteristic.battery_level.xml`, accessed on 09.07.2018

The client-server communication between the Android application and the Python Django server is based on a RESTful API. The messages are sent using the HL7 FHIR Observation[7] resource. Examples in JSON encoding for sending labeled BT and unlabeled HR measurements from the client to the server as well as for the communication of the classification result back from the server to the client can be seen in Appendix B, Listings B.1, B.2 and B.3 respectively.

4.2.1 Android Application

The Android application represents the bridge between the Cosinuss° One sensor and the Python Django server. It is the tool to be used by the user for interaction with the vital sign classification system. The application provides its functionality through four main views, the initialization-, settings-, connection- and main view. In addition, a fifth view shows an about page with information about the application and why it was implemented. Figure 4.3 displays an activity diagram, graphically showing the activities to be performed by the application and the flow between the different views. Screenshots of the views can be found in Appendix C, Figures C.1 - C.8.

The initialization view is shown when the application is started for the first time. It is used to initialize several settings needed for usage of the application. This includes the generation of the patient identification based on the phones current milliseconds, to ensure that the identification is random, and if the measurements should be sent to the ML server. If so, the user has additionally to provide the Internet address of the ML server, the transmission interval of the measurements and if the measurements should be sent for training or classification purposes. Afterwards, if the Android application is used for a second time, this view will no longer be displayed. Instead, these settings can be changed in the settings view. Moreover, the settings view enables the selection of the MLA that should be used for classification of the vital sign time-series. Available algorithms are the ones mentioned in Section 3.1.3.

The view visible after normal start-up of the Android application, *i.e.* not for the first time, is the main view. Because there exists no connection yet to the Cosinuss° One sensor, there will nothing be visible except a *please connect to the sensor* message. Therefore, the connection view has to be started using the menu. This view shows nothing except a spinning wheel, but in the background the BLE connection to the sensor is being established. After the connection has been initialized, the user is redirected back to the main view.

[7] `https://www.hl7.org/fhir/observation.html`, accessed on 09.07.2018

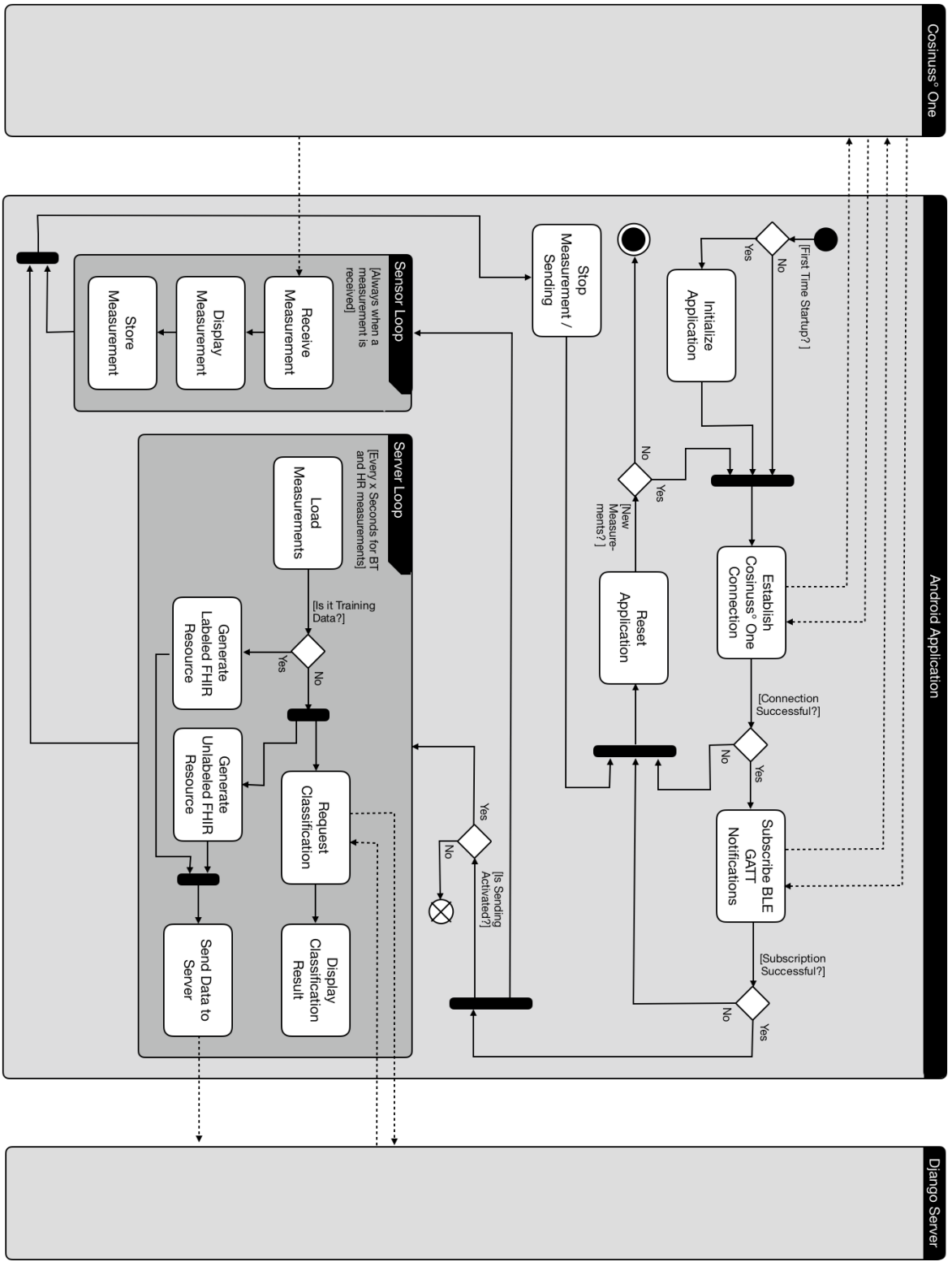


FIGURE 4.3: Activity diagram of the Android application.

Next, the user has to press the start-reading button. This initializes the subscription for the Reading Changed Notification Service of the Cosinuss° One sensor. It means that the sensor will send a message containing the BT or HR readings only if the measured values have changed. If such a message is received, the BT and HR display will be changed according to the new measurements. Additionally, the received measurements will be stored for later use, when sending them to the ML server.

The remaining part of the main view depends on the previously set settings. If sending of the BT and HR measurements to the ML server is activated, a background server service for sending the measured vital signs to the ML server will be started after connection setup to the sensor has finished. This service fetches the BT and HR values previously stored, uses a factory design pattern to build the HL7 FHIR Observation resources and sends them to the ML server using its RESTful API. This process is regularly triggered according to the transmission interval defined in the (initial-) settings. Additionally, if the data is sent as training data, two dropdown fields are shown in the main view to label the data. The selected labels are then added to the HL7 FHIR Observation resources during their creation as part of the interpretation data field. This is needed for the evaluation of the Cosinuss° One sensor and the MLAs as explained in Sections 3.1.2 and 3.1.4, and could be removed for productive use of the system. In contrast, if the classification option is enabled in the settings menu, the server service will send a classification request to the server in addition to the unlabeled HL7 FHIR Observation resources. This request is sent in the same interval as the vital sign measurements are being sent. The result of this request is another HL7 FHIR Observation resource containing the classification result in its interpretation data field. The result will then be extracted and displayed.

Pressing the stop-reading button in the main view resets the application for future use. It terminates the BLE connection to the Cosinuss° One sensor and stops the server service for sending the vital sign measurements and classification requests to the ML server.

4.2.2 Machine Learning Server

The Python Django server is the heart piece for the ML activities of the developed proof-of-concept prototype. It follows a web-based REST architecture, consisting of four major parts including a RESTful API, a storage service and two ML components as the training- and the classification service. The individual components are described in depth in the following paragraphs. A visualization of the architecture can be seen in Figure 4.4.

The RESTful API exposes a client's access points to the different services offered by the ML server using a facade design pattern. There are three different kinds of access paths, each corresponding to a different requestable service:

- **/store/observation/labeled:** HTTP POST request containing a HL7 FHIR Observation resource including the measured vital sign to be stored for training purposes in comma-separated value (CSV) formatted files.
- **/store/observation/unlabeled:** HTTP POST request containing a HL7 FHIR Observation resource including the measured vital sign to be stored for classification purposes in a SQL database.
- **/classify/<mla-type>/<patient-id>:** HTTP GET request to classify the last received measurements based on the provided MLA type *<mla-type>* and the provided patient identification *<patient-id>* by the client. Abbreviations for the MLA types *<mla-type>* which are supported by the system are:
  - *if* for the Isolation Forest algorithm
  - *lof* for the LOF algorithm
  - *ocsvm* for the OC SVM algorithm
  - *ae* for the Autoencoder ANN algorithm

The storage service receives from the RESTful API the HL7 FHIR Observation resource sent from the client. In a first step, the FHIR processing unit extracts the type of vital sign observation and the corresponding relevant information from the received HL7 FHIR Observation resource (*i.e.* type BT or HR and the associated patient identification, measurement timestamp, measurement value and unit, and for labeled data activity- and location information). If the data is marked as training data (*i.e.* it is received over the labeled path), it is transformed into CSV form by the CSV mapping unit and stored for each patient individually in a file based on the measurement type. This storage separation of BT and HR measurements is done because BT and HR measurements are received in different POST requests. Each request can only hold one type of measurement. This also applies for the later explained SQL storage of unlabeled data for classification purposes. Moreover, this unit creates for each measurement series a new CSV file to not have abrupt changes between single measurements (*e.g.* if one measurement series stops after doing sports and the next starts sitting before the television). A new measurement series is started if the last received measurement was more than five minutes ago. Five minutes were selected as threshold because the sensor sometimes does not send any information for a short time. This can be the case, for example, if the connection was interrupted in the elevator although the measurement series was not terminated.

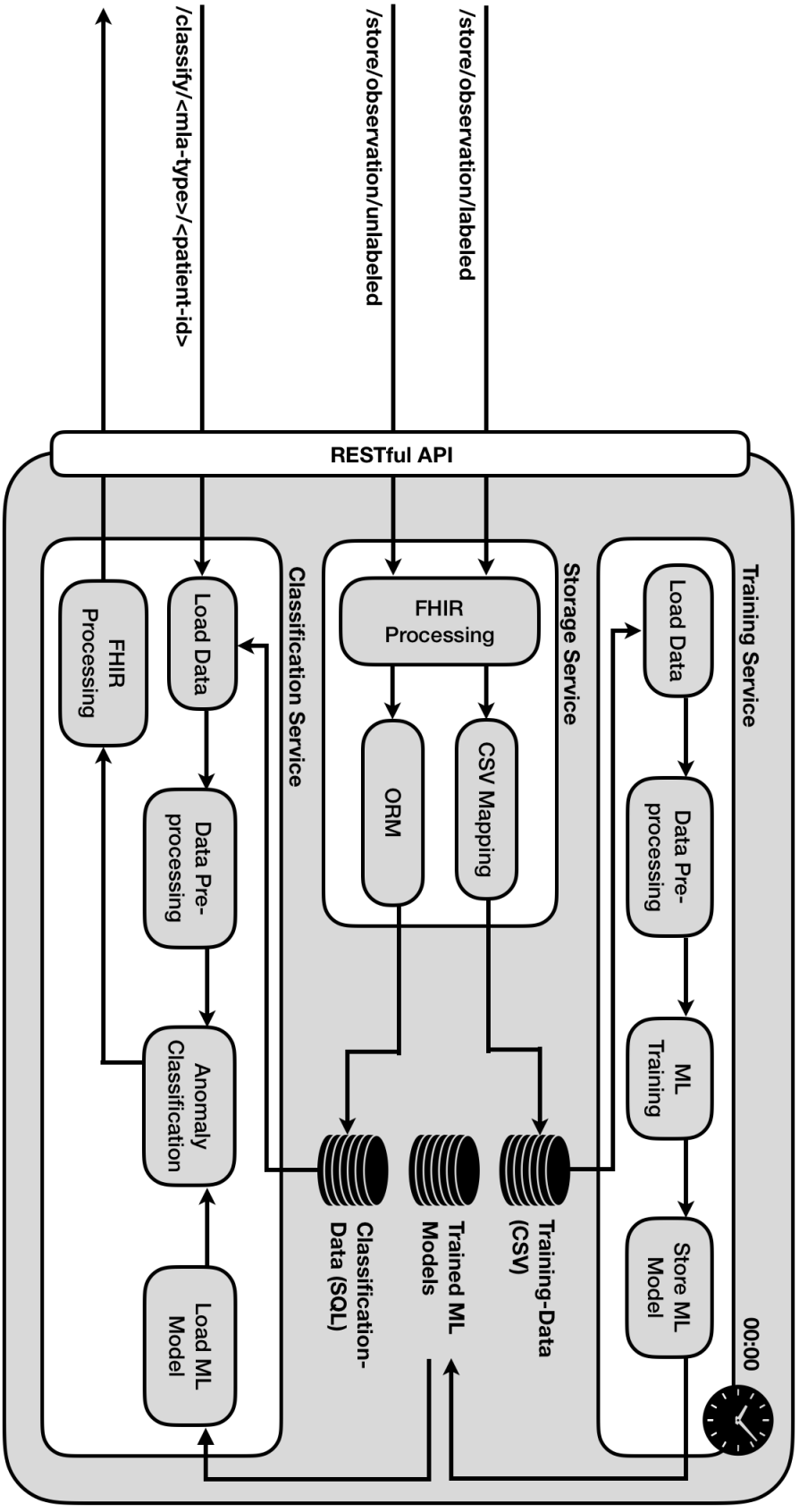


FIGURE 4.4: Server architecture implemented in the proof-of-concept prototype.

In contrast, if the data is sent for classification purposes (*i.e.* it is received over the unlabeled path), it is transformed by the object-relational mapping (ORM) unit into relational form to match the expected format of the SQL database. Similar to the CSV storage, there exist two different tables, one for BT and another for HR data. They store the same relevant information of the resources as the CSV storage, just without a label. Note that a single measurement is only stored if its value is above a threshold of 30 °C for BT and 50 bpm for HR. This reflects a plausibility assessment by filtering data which is not feasible (*e.g.* due to measurement errors of the sensor).

This type of storage for measured data based on two databases is done to have as few differences as possible between the proposed architectural concept and the implemented proof-of-concept prototype. It allows to have a clear distinction between training and not-training data. It reflects the proposed architectural concept, where there would be no classification data on the server after finishing the training data collection phase. The use of CSV based storage for the training data simplifies its later processing, because the Python libraries used for the ML part have special input functions which can process CSV files very efficiently. Similarly, using a relational database as storage for the classification data facilitates the efficient loading and generation of a single time-series based on the last generated measurements.

The training service is scheduled regularly for each patient on a fixed time depending on the server settings. For the proof-of-concept prototype it was decided that the training of the MLAs should always be executed daily at midnight. This is reflected in Figure 4.4 by the clock in the right upper corner. First, the load data unit fetches all the CSV files containing the BT and HR measurements of one patient and loads their measurement values. Then, the data processing unit is responsible for the preparation of the measurements for actual training and the generation of the vital sing time-series. Therefore, this unit merges the BT and HR measurements based on their timestamps, applies a gaussian smoothing to reduce single deflection measurements and normalizes the BT and HR measurements into the same interval. This is followed by the generation of time-series chunks for training of the different classifiers. The last step of the training preparation is to supplement the time-series with the mean and standard deviation of the individual BT and HR series, and the hour and minute of the recording time (middle of the time-series). This represents a data-driven approach for the analysis of the time-series which is supplemented by four statistical features. Therefore, the MLA input space consist of 56 dimensions (25 values of BT and HR measurements respectively plus four statistical features). Finally, the actual training of the four MLAs is executed in the ML training unit, using the grid search method for parameter tuning, before the store ML model unit safes each model according to the patient identification and the best parameters.

The classification service is triggered by the request of the mobile application. First, the load data unit fetches the last recorded BT and HR measurements based on the provided patient identification in the request path. The number of loaded measurements depends on the chosen length of a time-series in the training phase (25 values for each vital sign type in the implemented prototype). After that, the data processing unit executes the same data processing steps as in the training service component. The only difference is that just one time-series of the last recorded measurements is created. At the same time, the load ML model unit fetches the trained ML model from storage, according to the specified patient identification in the request path. This allows the anomaly classification unit to analyze the last generated time-series and to classify it as either normal or abnormal. Finally, the classification result is added to a newly generated HL7 FHIR Observation resource by the FHIR processing unit and returned to the client.

Chapter 5

Results

This chapter reports the results obtained after executing steps (vi) and (x) listed in Section 3.1. It first presents in Section 5.1 the evaluation of the Cosinuss° One BT measurements by showing its measurements in different surroundings and by comparing them to a commercially available digital thermometer. After that, the classification results of the different applied MLAs are shown in Section 5.2, using a confusion matrix to derive for each algorithm overall accuracy, sensitivity and specificity. In addition, the classification results for each type of abnormal data are displayed separately.

5.1 Cosinuss° One Body Temperature Measurements Evaluation

In order to evaluate the BT measurement quality of the Cosinuss° One sensor, its measurements were first verified for their general validity according to the normal range of human BT. Afterwards, the recordings were compared with those of a commercially available digital thermometer as it can be bought in every pharmacy. The aim of this comparison was to examine whether changes in BT could be recorded in the same ratio during a non-active and active state. Since the evaluation of the Cosinuss° One sensor was conducted by only one subject (N = 1), the following results must be treated with caution. They are visualized in Figures 5.1 and A.2.

5.1.1 General Validity of Measurements

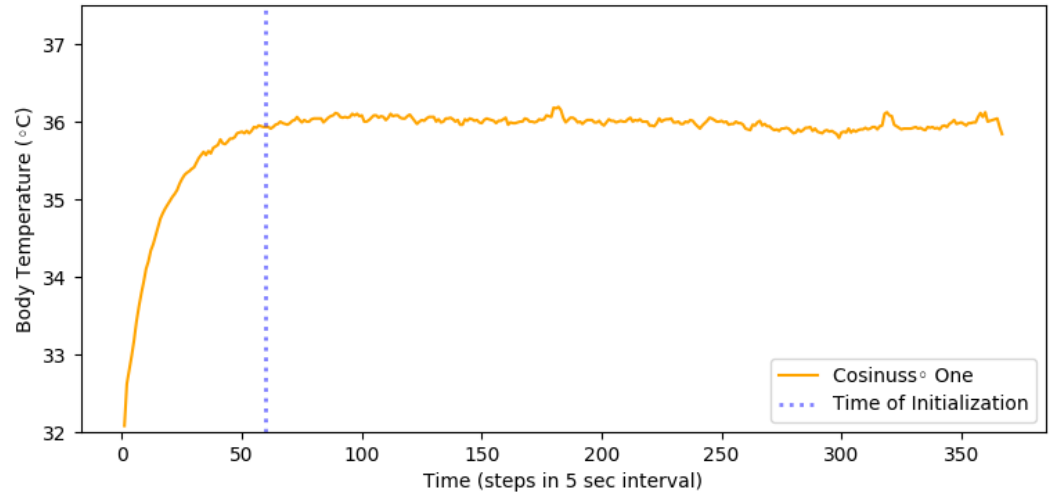
Figure 5.1a shows an example in which the Cosinuss° One sensor needed approximately four to six minutes, depending on its environment, until it was fully initialized and measured BT in a normal range. After initialization, the measurements stabilized with slight

fluctuations (around 36 °C in this example), provided that the subject was surrounded by not rapidly changing conditions. This type of initialization could be observed every time the sensor was turned on. However, the reached steady level varied depending on the environment.

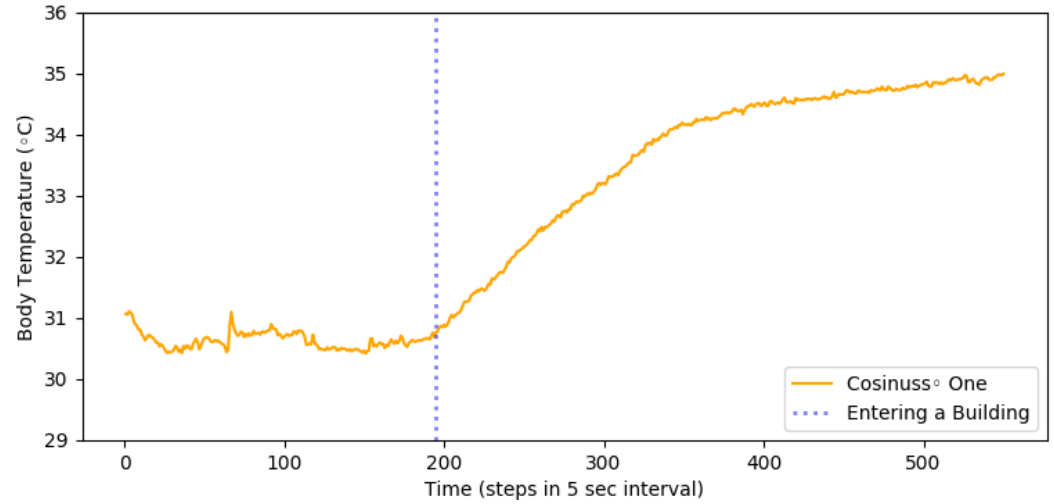
The influence of the surroundings on the measurements of the Cosinuss° One sensor can be seen in two further examples. Figure 5.1b displays approximately 45 minutes of measurements. The first third (∼15 minutes) was taken outside a building at 8 °C outside temperature, showing a BT fluctuating between 30.4 °C and 31.1 °C. The remaining measurements (∼30 minutes) were then taken inside a building. It can be observed that after the subject entered the building, the BT rose rapidly until 34.2 °C. Later, the curve became more shallow and the BT increased more slowly but still continuously until the end of the measurement series, where the Cosinuss° One sensor showed a BT of 35.0 °C. Surprisingly, this slow increase in BT could be observed during many measurements in controlled and constant environments like inside a building.

Furthermore, outside measurements were always quite inaccurate and the measured BT was strongly dependent on the surrounding temperature and weather (*i.e.* how warm or cold it was and whether it was raining, cloudy, windy or sunny). However, BT below 30 °C was never measured. The dependency of the measurement accuracy on ambient environmental factors can be seen in Figure 5.1c. It shows approximately 52 minutes of measurements on a sunny day during an outside temperature of 18 °C. The first quarter was measured inside a building, to create a steady baseline, recording a temperature of 36.2 °C with little fluctuations around this point. After the subject left the building and was standing in the shadow, the BT constantly dropped to 34.8 °C. Then, the BT rose sharply again to 36.0 °C when the subject changed his location, sitting in the sun for the following five minutes. After leaving the sun, the same sequence as before could be observed. At first, after the subject entered the shade for the second time, the measured BT fell even more to 34.6 °C, only to start rising again when it returned to the building. Remarkable is that the decrease in the shadow has the same tendency and velocity both times.

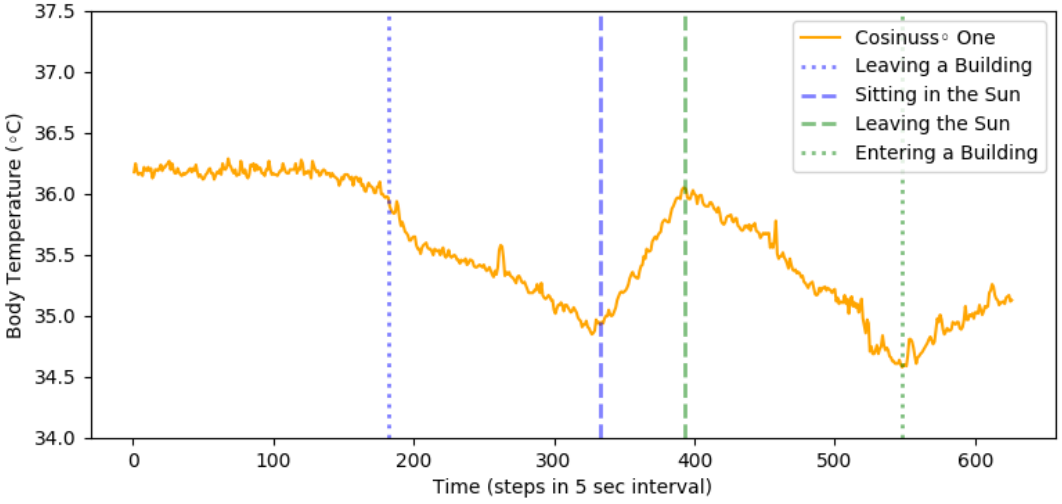
Dependencies to ambient environmental temperature could also be determined within a building. The measurements of the Cosinuss° One sensor were influenced by whether heating or air conditioning was turned on or whether a window was opened. Measurements at the same time and place, but on two different days with different indoor factors, could show differences of up to 1.5 °C.



(a)



(b)



(c)

FIGURE 5.1: Body temperature measurements made by the Cosinuss° One sensor: (a) initialization time of the sensor, (b) comparison between measurements inside and outside of a building, (c) influence of the weather on measurement accuracy.

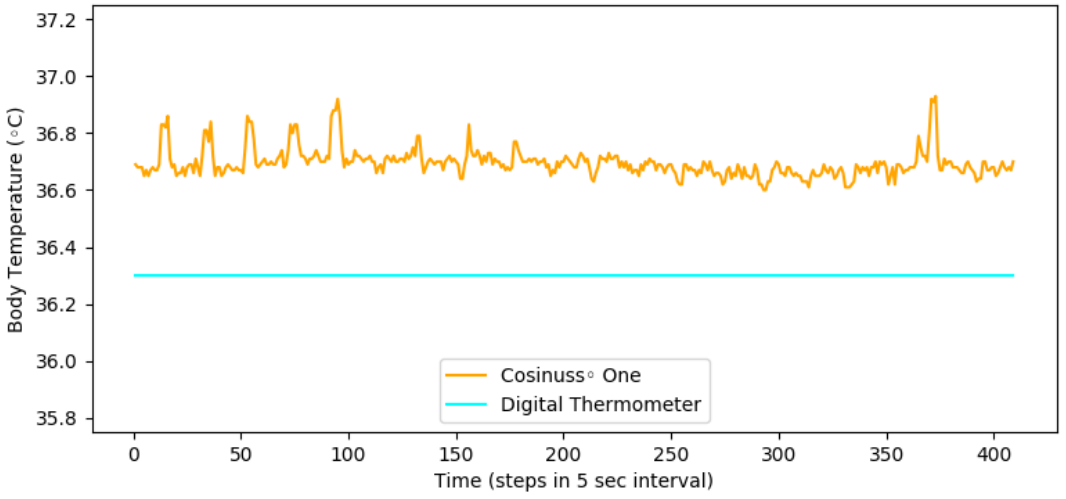
5.1.2 Comparison with a Digital Thermometer

The first comparison of the Cosinuss° One sensor with the digital thermometer was done in a non-changing environment, sitting on a chair inside a building for approximately 30 minutes. The measurements are visualized in Figure 5.2a. The Cosinuss° One sensor measured an average value of 36.8 °C and fluctuated by 0.3 °C from 36.6 °C to 36.9 °C. In comparison, the digital thermometer showed a constant value of 36.3 °C, which is 0.5 °C degrees lower than the average measurement of the Cosinuss° One sensor. Both values are in the normal range of human BT, having the same tendency. The differences of 0.5 °C could be explained by the different measurement techniques at different body positions.

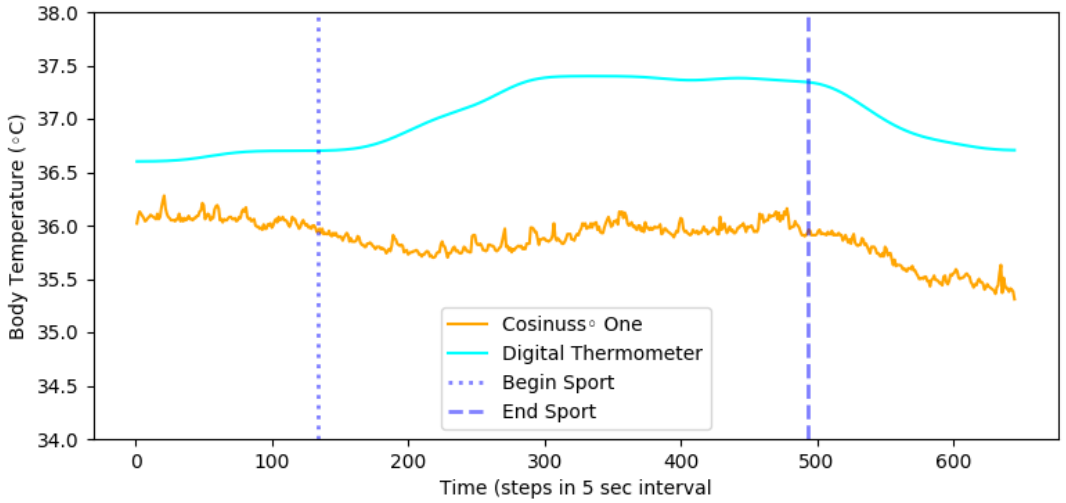
Figure 5.2b shows a second comparison during approximately 50 minutes, 30 of them the subject was in an active state on an elliptical trainer. In contrast to the previous comparison, the Cosinuss° One measurements were lower than those of the digital thermometer. It can be observed that the measurements did not coincide, not even in their tendency. The digital thermometer recordings increased by 0.6 °C after the workout started, then stabilized as the subject began to sweat and fell back to their initial values after completion of the workout. In comparison, the BT measurements of the Cosinuss° One sensor decreased in the first part of the workout by 0.4 °C. Then, the measured BT increased during the second part until the initial values were reached, only to drop again significantly for 0.8 °C after the workout finished. While the measurements of the digital thermometer are described in similar ways as in literature, the measurements of the Cosinuss° One sensor are rather astonishing and could be explained by sweat development. A more in depth interpretation for reasons of the disparity between the two sensors can be found in Section 6.1.

5.2 Machine Learning Algorithms Evaluation

The results of the experiment to analyze the four MLAs are presented in two parts. The first part reports an overall evaluation of the algorithms, using all three types of anomaly data together as test set. Then, the results for each type of anomaly data are presented separately. As with the Cosinuss° One sensor, this evaluation was performed on data collected by only one subject (N = 1). Therefore, these results must be treated with caution as well.



(a)



(b)

FIGURE 5.2: Comparison of body temperature measurements between the Cosinuss° One sensor and a commercially available digital thermometer: (a) without activity while sitting on a chair, (b) during sports on an elliptical trainer.

5.2.1 Overall Results

The overall evaluation results of the different MLAs applied to the vital sign time-series measured by the Cosinuss° One sensor can be seen in Table 5.1. All algorithms reached an accuracy higher than 80 %. The best result was obtained by the OC SVM with just under 90 %. The LOF and the Isolation Forest follow with almost similar results, being just 1 % respectively 2 % less accurate. The Autoencoder achieved the worst overall classification results with 82 % and therefore almost 8 % less accuracy. If only sensitivity is considered, which shows if all anomalies are detected, the Isolation Forest achieved the best result with just over 90 %. The OC SVM and the LOF were almost equally good. The Autocoder once more achieved the worst result with 76 % sensitivity.

Table 5.1: Overall classification results of the different selected machine learning algorithms.

		Local Outlier Factor		Isolation Forest		One-Class SVM		Auto-encoder	
		−1	**+1**	**−1**	**+1**	**−1**	**+1**	**−1**	**+1**
Confusion Matrix	**−1**	142	22	148	16	144	20	125	39
	+1	14	150	24	140	13	151	19	145
Accuracy		89.02 %		87.80 %		89.94 %		82.32 %	
Sensitivity		86.59 %		90.24 %		87.80 %		76.22 %	
Specificity		91.46 %		85.37 %		92.07 %		88.41 %	

Regarding the specificity, which is responsible for low false alarm rates, best results were received again by the OC SVM with 92 %. The high sensitivity of the Isolation Forest resulted in the worst specificity of all algorithms with 85 %. In conclusion, comparing the four MLAs across all three evaluation methods, the OC SVM clearly performed best. In the midfield is the LOF, which obtained only slightly better results than the Isolation Forest. The Autoencoder performed worst.

5.2.2 Results Based on Anomaly Types

To investigate which type of error was caused by which type of abnormal data, the accuracy, sensitivity and specificity of each algorithm was evaluated a second time in relation to each type of abnormal data. The specific results can be seen in Table 5.2. For the type sport, all algorithms performed very well and achieved the best results, reaching accuracy scores of 93 % and above. For sensitivity, three of the four algorithms reached 100 %. Only the LOF was a little worse at 98 %. Specificity showed also good results. The worst algorithm, the Isolation Forest, reached 86 %. The best specificity was obtained by the OC SVM with 92 %. Over all three evaluation types regarding the abnormal time-series sport, the OC SVM performed with the highest scores. The lowest scores were achieved by the Isolation Forest.

The results differed for the other two types of anomalous time-series measurements. The OC SVM and the Isolation Forest scored better results for type metro whereas the LOF and the Autoencoder scored better for the anomaly class eating.

For the type metro, the highest accuracy was achieved by the Isolation Forest with 91.35 %, the lowest accuracy by far from the Autoencoder with 72 %. The Isolation

Table 5.2: Detailed classification results of the machine learning algorithms split according to the selected type of abnormal data.

			Sport		Metro		Eating	
			−1	**+1**	**−1**	**+1**	**−1**	**+1**
Local Outlier Factor	**Confusion Matrix**	**−1**	63	1	37	15	42	6
		+1	6	58	5	47	3	45
	Accuracy		94.53 %		80.77 %		90.63 %	
	Sensitivity		98.44 %		71.15 %		87.50 %	
	Specificity		90.63 %		90.38 %		93.75 %	
			−1	**+1**	**−1**	**+1**	**−1**	**+1**
Isolation Forest	**Confusion Matrix**	**−1**	64	0	51	1	33	15
		+1	9	55	8	44	7	41
	Accuracy		92.97 %		91.35 %		77.08 %	
	Sensitivity		100 %		98.08 %		68.75 %	
	Specificity		86.15 %		84.62 %		85.42 %	
			−1	**+1**	**−1**	**+1**	**−1**	**+1**
One-Class SVM	**Confusion Matrix**	**−1**	64	0	45	7	35	13
		+1	5	59	5	47	3	45
	Accuracy		96.09 %		88.46 %		83.33 %	
	Sensitivity		100 %		86.54 %		72.92 %	
	Specificity		92.19 %		90.38 %		93.75 %	
			−1	**+1**	**−1**	**+1**	**−1**	**+1**
Auto-encoder	**Confusion Matrix**	**−1**	64	0	29	23	32	16
		+1	7	57	6	46	6	42
	Accuracy		94.53 %		72.12 %		77.08 %	
	Sensitivity		100 %		55.77 %		66.67 %	
	Specificity		89.06 %		88.46 %		87.50 %	

Forest also performed best in terms of sensitivity with 98 %. The next best algorithm, the OC SVM, achieved 87 % sensitivity. The Autoencoder was the worst performing MLA with a very low sensitivity of 56 %. The OC SVM and the LOF achieved the highest specificity with 90 % each, followed by the Autoencoder with 88 % and the Isolation Forest with 85 %. The OC SVM was again the best-performing algorithm over all three evaluation methods, closely followed by the Isolation Forest. The LOF obtained only slightly worse results. The Autoencoder reached the lowest scores.

Last but not least, the results for the anomaly type eating. The best results regarding accuracy were achieved by the LOF, being slightly better than 90 %. The worst result were recorded by the Isolation Forest and the Autoencoder with 77 % each. Regarding sensitivity, the LOF performed again as best algorithm, the others being all at least 14 % less sensitive. In the case of specificity, the LOF had the best results together with the OC SVM, each having a specificity of almost 94 %. The Isolation Forest and the Autoencoder reached a specificity of 85 % and 88 % respectively. Overall, the LOF performed for the type eating a little better than the OC SVM. Even though the Autoencoder achieved better results with the type eating than with the type metro, it was still the worst of all four algorithms. But, the results of the Isolation Forest were just slightly better.

Chapter 6

Discussion

The goal of this thesis was to asses the hypothesis if *a real-time body temperature and heart rate monitoring system enables personalized classifications of physiological response patterns as either normal or abnormal*. For this reason, five primary objectives were defined and implemented. This chapter will analyze, interpret and then in detail discuss the results presented in the previous two chapters with regard to the fulfillment of the these objectives. Then, a final conclusion will be drawn in which the hypothesis will be accepted or rejected.

To begin with, it should be said that the evaluations conducted during the implementation of the objectives were performed on one subject (N = 1). Therefore, the discussions of Sections 6.1 and 6.3 are only valid for this subject and no statistical significance can be derived.

6.1 Cosinuss° One Sensor

The first objective was to evaluate the measurement quality of the chosen Cosinuss° One sensor and therefore its suitability for real-time detection and classification of changes in vital signs. The Cosinuss° One sensor was chosen because of its supposedly comfortable wearing position in the ear, its long battery life, its affordability and most important its promised ability to measure BT and HR with high accuracy in one device.

Although the manufacturer promises a high wearing comfort due to the used materials, this could not be confirmed by the subject. In the first few hours after the sensor had been placed in the ear, it was comfortable and not disturbing in any situation. It was also often the case that the subject forgot that he was wearing the sensor. But, after wearing it for several hours ($>$ 4h), his ear began to hurt. Therefore, the sensor had

to be removed for at least 30 minutes to be able to wear it painlessly again. However, this is a personal opinion of the subject and could not be the case for others. But, it is not beneficial for continuous measurements. Furthermore, it was pleasing that the one hour indicated for charging the sensor was always adhered to. With its mid-tier price segment and if the sensor would not start to hurt in the ear after a certain amount of time, the alternating use of two sensors could guarantee 100 % continuous recording of vital parameters in everyday life.

The most important selection aspect of the Cosinuss° One sensor was the promised high measurement accuracy by the manufacturer. For a system, as it was developed in this thesis, data quality is a critical parameter for the overall system performance. In other words, the classification accuracy of the MLAs can only be as good as the data used for their training. If the measured data is noisy or inaccurate, the accuracy, sensitivity and specificity of the selected algorithms will decrease. Since evaluations of Cosinuss° One HR measurements were already carried out in other reviews, which confirmed the high quality of this measurement type, this thesis only dealt with the quality evaluation of the BT measurements. The first observation made during the sensor evaluation is that the exact positioning of the sensor in the ear plays an important role in obtaining accurate BT measurements. It took the subject some practice until the sensor was positioned in the perfect place and led to possibly accurate measurements. Although the sensor is designed for usage during sports, strong movements could move the sensor slightly in the ear, leading to a significant decrease in measurement accuracy. This is not only the case for BT, but has also been observed in HR measurements. The additional grab rail, included in the delivery scope of the sensor, partially counteracted this.

The actual BT measurement evaluation of the Cosinuss° One sensor was then carried out in two steps. First, the measurements were assessed for their general validity before comparing them with those of a commercially available digital thermometer. It was noticeable that the Cosinuss° One sensor needs an initialization time of about four to six minutes to display measurements in a for a human normal interval. In contrast, the digital thermometer, which was used as a reference for evaluation, measured a feasible BT after about one minute of initialization. Furthermore, it was surprising how strongly the BT measurements were dependent on ambient temperatures. With a cold outside temperature around 8 °C or windy conditions, the measurements could fall as low as 31 °C. This is outside of a valid range and no human could withstand a BT this low. Conversely, direct solar radiation could increase BT measurements in regions associated with fevers. This could be normal for a human, especially during prolonged stay in very hot and humid regions as described in the literature. But, since the sensor was also sensitive to cold environmental temperatures and exposure to high temperatures was just for a short time, it is more likely that these changes were influenced by the environment

as well. Surrounding temperature changes in more stable locations, like inside a building caused by heating, air conditioning or an open window, could also influence the BT measurements of the Cosinuss° One sensor. Even though these fluctuations were still in the normal BT range of a human (36 °C - 37.5 °C), it is unlikely that they were just normal fluctuations. It is more plausible that they were also due to environmental changes.

In a second step, the measurements of the Cosinuss° One sensor were compared with measurements of a commercially available digital thermometer. This comparison showed big deviations. In a controlled environment within a building, while the subject was sitting on a chair, there was an average measurement difference almost 0.5 °C. But, both devices showed the same tendencies in their measurements. Additionally, the Cosinuss° One sensor detected during the measurements strong fluctuations in short time intervals, which are not feasible. However, some of the difference between the two sensors could be attributed to the different measuring methods and positions.

During the second direct comparison, while the subject was exercising on an elliptical trainer, there was an even bigger difference with no tendency similarities. The BT measurements of the digital thermometer increased significantly after beginning to exercise, stabilized after the development of sweat and then returned to the initial values after finishing the workout. This corresponds to patterns as described in the literature. In contrast, the BT measured by the Cosinuss° One decreased during the first part of the workout. In the second part, the BT rose again to the initial values, only to drop one more time after the workout ended. A possible explanation for these measurements could be related to development of sweat. At the beginning of the workout, *i.e.* when the subject starts to sweat and thus sweat is present on the skin, the measured BT drops in combination with the surrounding cold temperature. This is then compensated by the physical activity, increasing the internal BT of the subject. A similar change can be observed after the workout. Sweat is still on the test person's body and therefore cools it, but physical activity is no longer available for compensation. As a consequence, the BT drops again.

As a conclusion, it can be said that the Cosinuss° One sensor did not measure BT accurately for the subject. Its measurements were strongly influenced by the environment in which they took place. Since measurements in same environments behaved mostly with the same tendency, *i.e.* had the same inaccuracy, the measured BT of the Cosinuss° One sensor can still be used for anomaly detection. Measurements during the deterioration of a patient's health will not reflect the correct BT, but will still be different from measurements in a healthy state of the patient.

6.2 Proof-of-Concept Prototype

The second and third objectives of this thesis were to develop an architectural concept for a distributed real-time monitoring and classification system of vital signs and to implement this concept as a proof-of-concept prototype.

The designed architectural concept proposes three major components, a sensor for measuring selected vital signs, a ML server for storage of training data and continuous training of the MLAs in a person-related manner, and a mobile application for classification of new measurements. Additionally, the mobile application acts as a bridge between the two other components and the user. The concept was designed as a multi-tier distributed system, where the connection between sensor and mobile application is built as a point-to-point architecture. This design was chosen because the communication between the sensor and the mobile application is of private nature and does not have to include other devices. The communication between the ML server and the mobile application was designed as client-server architecture, since the ML server provides his service to multiple mobile applications. The use of a sensor and a mobile application in the system design allow continuous recording and monitoring of PRPs corresponding to activities, diseases and the environmental context, as the mobility of this part of the system is guaranteed. The mobile application also displays the recordings and results of the MLAs live to the user, which allows him to track his data. Outsourcing the storage and training of the MLAs to a ML server brought the advantage that the most complex part, in terms of computation, is not carried out on the mobile application itself. This saves storage space and battery power on the device. The disadvantage is the necessary Internet connection for uploading the measured training data to the server and for downloading the trained ML models to the mobile application. But, after the model has been downloaded to the application, future new data can be classified without Internet connection. Since the initial phase of training data collection is much shorter than the later continuous classification phase, this trade off between required Internet connection and saving storage respectively computational resources could be accepted. As a consequence, this approach eliminates one of the biggest sources of errors, making classifications possible as long as the device and the sensor have battery power. Therefore, the conceptualized system has a high resilience against network connection failures.

In addition, the conceptualized system should allow the user to include measurements into the training process which led to false positive classifications, in order to reduce future false alarms and to increase accuracy, sensitivity and specificity of the algorithms over time. This represents a countermeasure to alarm fatigue. Using a cache for false positive classification data when there is no Internet connection available, ensures that

false positive measurements are not lost and can be sent later to the ML server. This enables continuous learning if the MLAs are retrained in equal intervals.

The three components of the architectural concept were realized by the proof-of-concept prototype using a Cosinuss° One sensor, an Android application and a Python Django ML server. During their implementation, great importance was assigned to select and apply already established but modern technologies in order to make the system as flexible and modular as possible, trying to minimize future extension and maintenance efforts:

- In addition to the above-mentioned advantages, the Cosinuss° One sensor offered a further one in terms of implementation. Its open source BLE API simplified the integration into the proof-of-concept prototype. The API, based on standardized BLE GATT profiles, made it possible for the Android application to connect and read the measured vital sign values effortlessly.
- Using Android as the host OS for the mobile application had the advantage that a versatile SDK and specialized libraries for BLE and REST message exchange were available to implement the tasks of the application.
- Python and its libraries for development of the ML server made fast prototyping possible. Due to following the REST architecture paradigm, modularity increased while complexity was lowered, leading to a desired loose coupling between system components. An example for this would be that the database for storage of the vital sign measurements can be changed without changing any other part of the system. Furthermore, using a REST architecture in the ML server component made the system more reliable, portable, scalable and efficient. As a result, the ML server supports very light weight clients, whose functionality can be increased by the server using code transfer. One example is sending the trained ML models to the mobile phone application as proposed in the architectural concept.

Two important parts of the initial concept were not realized by the proof-of-concept implementation. On one hand, the models of the trained MLAs were not transferred to the Android application. The classification of new measurements after the training phase takes also place on the server component. For this to work, the classification data had to be sent to the server, as it was done with the training data, but stored separately. On the other hand, the user feedback was not implemented. These two missing features were omitted for time and simplicity reasons, since they had no influence on assessing the hypothesis. However, for a later productive use of the system this would have to be added in order to offer the best possible platform to users.

Additionally, by deploying the prototype on a Raspberry Pi, it could be shown that the system can be set up at everyone's home for little cost. The single-board computer, which has a low computing power compared to normal servers, could be utilized because the used libraries favor efficient calculations and the implemented REST architecture supports scalability. Using this approach, a user has full self-determination over the system and the collected data. The data is not stored on third party servers.

6.3 Machine Learning Algorithms

The last two objectives were to prove that MLAs can be used to learn the individual PRPs of a person's vital signs, taking into account activities, demographic factors and the environmental context, and thus can classify a change in tendency of vital signs as physiologically normal or abnormal.

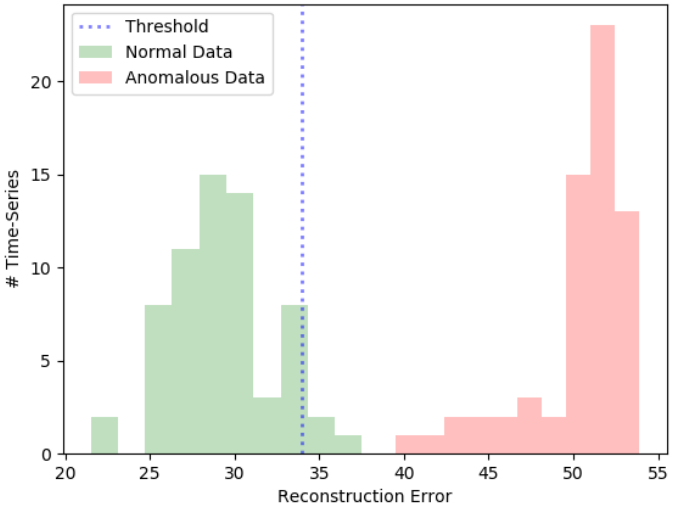
To be able to achieve these two objectives, the algorithms LOF, Isolation Forest, OC SVM and Autoencoder were selected, added to the system and evaluated on the basis of data collected from one subject. When selecting suitable algorithms for anomaly detection, great importance was attached to the fact that they are derived from well known and frequently used MLAs, which differ in their mathematical foundations. The biggest advantage of all these algorithms is that they consider great inequality in the distribution of the data. This means that anomalies occur only as a small fraction of all data. In contrast, other MLAs often make the assumption that the different possible classes in the data are evenly distributed. A disproportionately large represented class can then lead to biased classification results.

In order to test the performance of the algorithms, a baseline of vital signs was recorded as training data for three consecutive days (72 h). By recording for this amount of time, it could be ensured that enough training data was available and that the curse of dimensionality could be avoided. The recorded data was labeled according to the performed activity and location. Since irregular vital sign data cannot be generated on purpose, the data recorded during sports, in the metro and while eating were later removed from the training data and regarded as artificial anomalies. To train the algorithms, the training data was then split into chunks of two minutes with an overlap of 30 seconds. The input to the MLAs was based on a data driven approach and supplemented by four statistical features, since all time series have the same length and fast analysis in health data is crucial. A data-driven approach was only possible because the number of values in one time-series was not very high. If time-series with more measurements would be used, a more statistically driven approach or the generation of more training data would have to be considered to avoid the curse of dimensionality. Then, after training the MLAs using

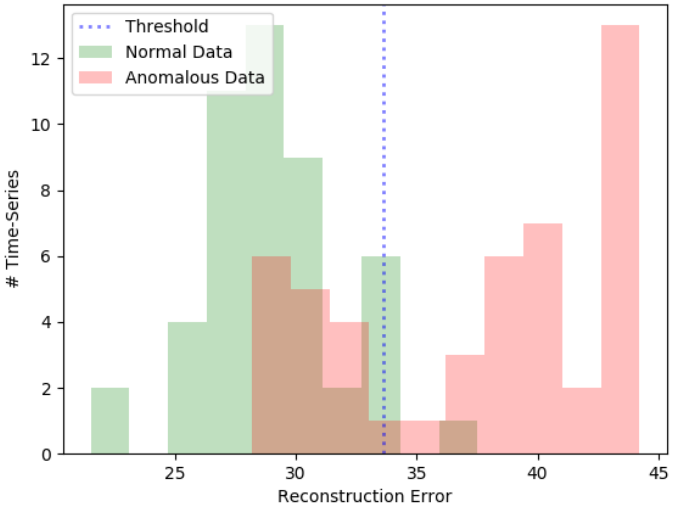
the grid search method, the accuracy, sensitivity and specificity of each algorithm was evaluated. All four algorithms achieved good results with accuracies between 82 % and 90 % for the subject. The OC SVM performed best with an accuracy of 89.94 %. For the individual types of artificial anomalies, the best results were obtained for the type sport. All algorithms achieved an accuracy of over 93 %. These results can be attributed to fast noticeable changes in BT and HR after starting an exercise session, which can even be felt by the performer. The literature describes that during physical exertion, HR increases and a person starts to sweat to regulate BT. The good results for sports were to be expected and show that the algorithms can learn PRPs. In the other two categories of artificial anomalies, the changes are not obvious to a person, whereas for the algorithms they are. They could classify these types as artificial anomalies, but with lower accuracy than during sport. Whether the type metro or eating was better detected, depended on the algorithm. However, results were always higher than an accuracy of 75 %, except in the case of type metro using the Autoencoder. The exact results, also for sensitivity and specificity of the MLAs, can be found in Table 5.2. A reason for the worse results for the activities eating and metro, compared to the activity sport, are because of the smaller difference between the measured BT and HR values in relation to normal data. In addition, the transition phase for adaption of the vital sings to the new situation is longer. Missclassifications were mainly caused by boundary measurements between activity changes. It can therefore be assumed that correct classifications will occur with delay. Another reason for correct but delayed classification can be traced back to the selected time frame of two minutes for the analysis. For an anomaly to be detected as such, an algorithm needed at least half of all measurements of one time-series in an anomaly state. This would correspond to a minimum delay of at least one minute until classifications change between normal and abnormal or vice versa. This could limit very time critical medical anomaly detection, especially when the analyzed time frame is increased. The reason why the Autoencoder performed worst in all categories could be due to the fact that no special layers were used. Better results could probably be achieved using more sophisticated ANNs.

It is worth noting that the use of time in addition to measurements during training had little impact on classification results. It is true that if abnormal data is always collected at the same time, it can be detected more easily and the classification results increase. This was taken into account during the experiment and the abnormal data was not collected every day at the same time.

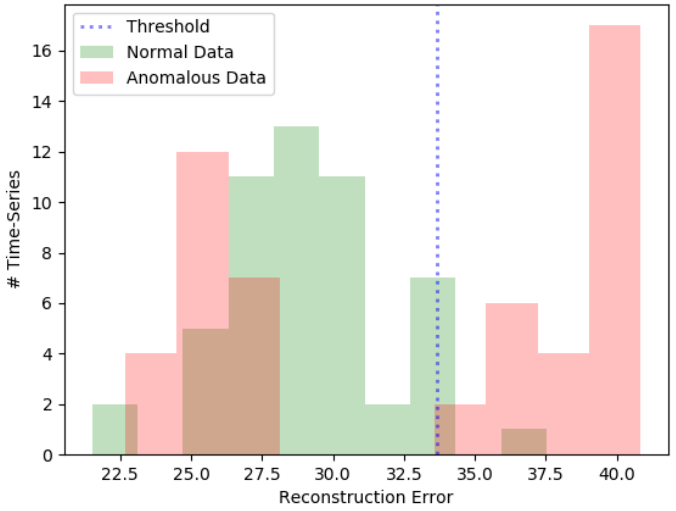
Deciding whether a measurement time-series is normal or should be examined more closely, *i.e.* because it is regarded as abnormal, is very important in medical fields. Many false positive anomaly detections can lead to the problem that physicians and nurses no longer trust those systems and thus develop alarm fatigue. When making



(a)



(b)



(c)

FIGURE 6.1: Comparison of the reconstruction error distribution for normal and the three types of anomalous data with reference to the calculate decision boundary: (a) sport, (b) eating, (c) metro.

decisions in an automated fashion, they are mostly dependent on the selected threshold for the algorithms. This threshold can be used to adapt an algorithm to its specific task and therefore to change its sensitivity and specificity. But, an increasing specificity of an algorithm often has the undesirable effect that its sensitivity decreases. The same applies the other way around. This is visualized in Figure 6.1. It shows the reconstruction error of the Autoencoder for normal and the three types of abnormal data in comparison to the selected threshold. Shifting the threshold in Figure 6.1a to the right, by increasing the acceptable error, would increase specificity. Sensitivity on the other hand would not change because the threshold would move between the two different data types. Shifting the threshold in Figure 6.1b to the left would decrease the specificity and increase the sensitivity, therefore leading to more undesired false alarms. However, it would insure that all anomalous data is recognized. Changing a threshold can also have no desired effect. In Figure 6.1c, to improve specificity, the threshold would have to be shifted to the left by such an extent that the sensitivity would be close to zero. In contrast, shifting the threshold to the right would not have much effect on sensitivity. It would further worsen specificity. For the other three algorithms the threshold is calculated automatically during training to maximize accuracy. If sensitivity or specificity should get higher weights than accuracy, the threshold has to be calculated separately.

Another problem when training MLAs is overfitting. This means that although the algorithm can classify the available training data very accurately, it fails for similar unknown data. This would lead to many false positive classifications in anomaly detection. A possible example of overfitting can be seen in Figure 6.2. It shows a fraction of a trained Isolation Forest. When not defining a maximal depth of a tree, the training process will isolate each training sample, building a very large tree, possibly overfitting the data. In order to solve this problem, the same amount of normal data as available abnormal data was not used in training. This data has been retained for evaluation. In this way, it could be ensured that similar normal data could be classified correctly later and that no or only a little overfitting is present.

From the performance results of the four MLAs and their subsequent discussion, it can be assumed that the two objectives of this part of the thesis have been fulfilled. The selected MLAs were successfully used for anomaly detection in vital sign time-series data. Since the algorithms have worked quite well, it can be assumed that for the same subject, other algorithms based on the same mathematical principles could also be applied to anomaly detection in vital sign time-series data. Possibly achieving better results. However, this still needs to be validated.

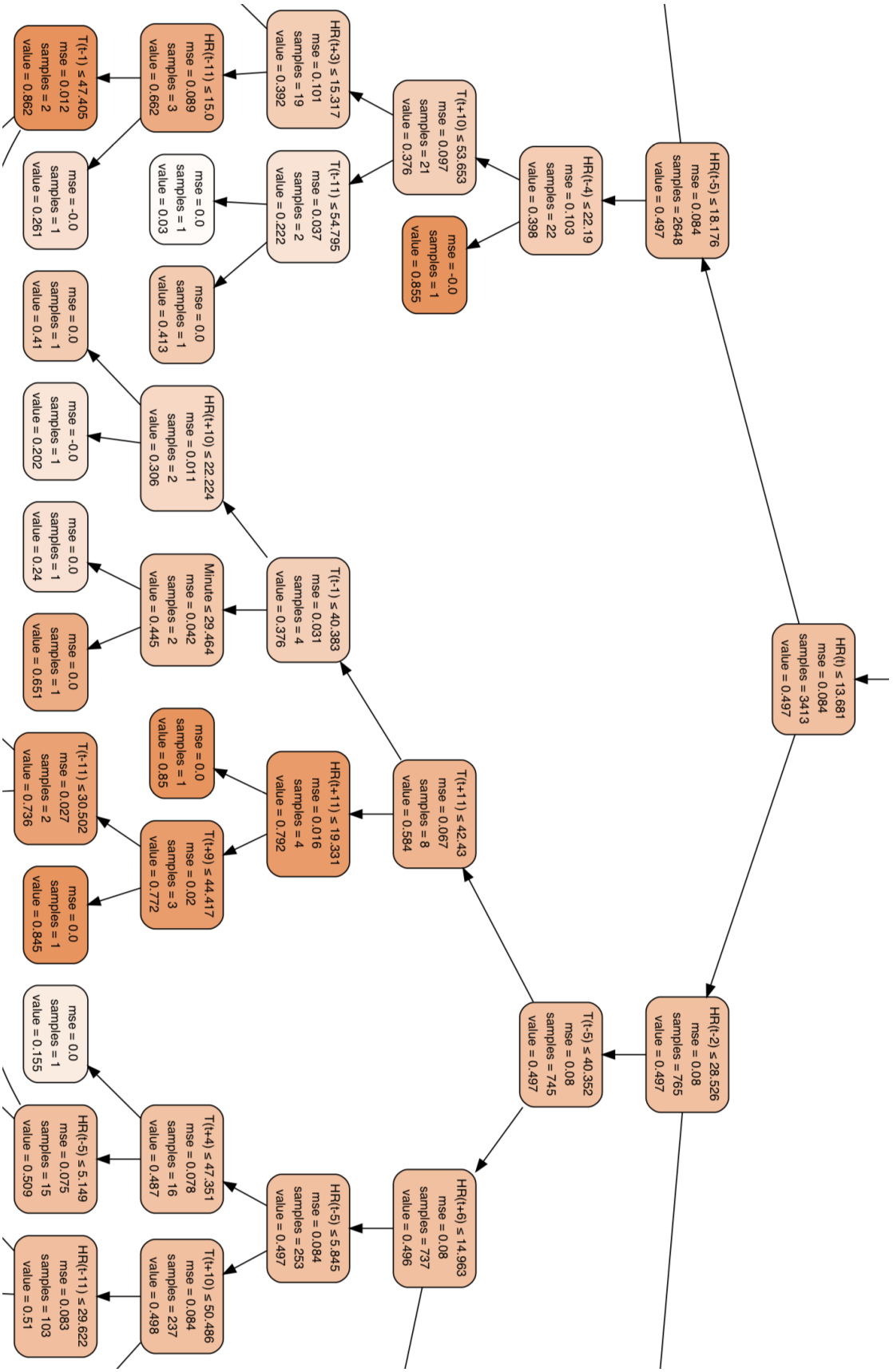


FIGURE 6.2: Excerpt from a single decision tree with possible overfitting of a trained Isolation Forest.

6.4 Conclusion

This thesis has taken a first step towards the development of a real-time classification system for anomalies in vital sign data in order to identify the deterioration of a persons health. The primary objectives of this thesis were the evaluation of the Cosinuss° One sensor, the development of an architectural concept for a real-time vital sign classification system, its implementation in a proof-of-concept prototype and the evaluation of its inherent MLAs. The selected MLAs for anomaly detection in vital sign time-series data were LOF, Isolation Forest, OC SVM and Autoencoder. These algorithms achieved an accuracy between 82 % and 90 %. The best result was obtained using the OC SVM with an accuracy of 89.94 %. Based on the results and the subsequent discussion it can be assumed that a real-time body temperature and heart rate monitoring system enables personalized classifications of physiological response patterns as either normal or abnormal. The hypothesis of this master thesis is therefore supported.

This academic work is subject to two main limiting factors. On one hand, due to a lack of resources, the experiments to evaluate the Cosinuss° One sensor and the MLAs were only conducted on one person ($N = 1$). In order to obtain statistically meaningful results, the evaluations have to be conducted during the same scenarios on a bigger sample size ($N > 1$).

On the other hand, the accuracy, sensitivity and specificity of the MLAs were only evaluated on regular medical data using different activities as the artificial anomalous counterpart. This does not imply that the system will work for diseases as well. More extensive testing would be needed to show that this real-time classification system of vital signs can detect the deterioration of a subjects health. Possibilities for approaching these limitations are discussed among other aspects within the following last chapter concerning the outlook.

Chapter 7

Outlook

In this last Chapter, the thesis will be finalized by an extensive outlook on future research topics. Since the real-time vital sign classification system was developed as proof-of-concept prototype, there are several ways in which this work could be continued to deploy it in a productive environment. The extension possibilities include the use of more data obtained by multiple other sources, changes in the architecture of the system, further analysis and application of different MLAs and the recorded data, and a more detailed overall evaluation of the system. These options are explored in more detail in the following sections.

7.1 Data Sources

As mentioned in the previous chapter, data sources are very important for a system like the one presented in this thesis. More sources and thus more available data for the detection of vital sign irregularities could improve the results of the MLAs. Therefore, the vital signs recorded by the Cosinuss° One sensor could be supplemented by additional medical and non-medical data sources. Besides BT and HR, other vital signs like respiratory rate or blood pressure could be recorded and used for anomaly detection. However, it could be difficult to find a suitable sensor that integrates discreetly into everyday life and therefore is not disturbing in a normal daily routine. Non-medical data could also be included in the analysis. Nowadays, almost every mobile phone has an acceleration- and location sensor. They could provide important data about the current situation of a user. For example, data from the accelerometer could be used to detect falls, or to trigger alarms if a patient is not moving at an unusual time. Another possible approach would be to investigate the influence of each data source and individual vital sign on the classification result.

7.2 System Extensions

For the use of the real-time classification system in a productive mode, the proof-of-concept prototype should be first adapted to the original architectural concept. In particular, porting the MLAs to the Android application, in order to prevent dependency on a Internet connection for classifications, would have priority. This would increase the resilience of the system. It would also be advisable to implement the user feedback to steadily reduce the false positive classification rate. Furthermore, the mobile application could be made available for other mobile operating systems like iOS.

Mobile phones can pose another problem. They often interfere with everyday life or are not at hand, because for example, they are stored in a pocket or a bag. One solution would be to port the mobile application to smart watches. They are less intrusive in everyday life and alarms can be communicated to the user even faster and more reliable, using different vibrations on the wrist depending on the severity of the alarm.

Another extension possibility of the system is concerned with the type of alarms displayed to a user in case of a possible anomaly. A single alarm could be extended by a series of alarms, applying different escalation levels. If the first alarm is not confirmed within a certain amount of time by the user, it could be relayed to family members or even to a medical facility in order to initiate the appropriate medical follow-up. Methods from the field of Ambient Assisted Living would be a useful starting point for this extension.

7.3 Analysis and Application of Machine Learning Algorithms

The part of anomaly classification using MLAs also leaves room for expansion. In order to further improve the accuracy, sensitivity and specificity of the selected algorithms, a more in-depth exploration and analysis of the measured data would make sense. Besides using the measured vital signs and the four calculated values for the classification of the time series, additional statistical features could be computed and incorporated. There are many features that would be possible including maximum and minimum or skewness, kurtosis, trend and seasonality. A good overview of important time-series features and a description of a Python library for their computation with the name tsfresh was published by Christ et al. [103].

Also, the selected algorithms could be improved. Instead of using the generated threshold value of the used scikit-learn library, it could be calculated manually for Isolation Forest, LOF and OC SVM. This would lead to more freedom and control over sensitivity and

specificity. The Autoencoder could be improved as well. At the moment a very simple ANN is used. More sophisticated ones like recurrent neural networks or convolutional neural networks (CNN) could improve the results of the Autoencoder. A successfull example of using a CNN for classification of time-series data is given by Mahabal et al. [104]. In order to classify their data, the initial time-series inputs were converted into two-dimensional image like representations, which enabled to use different CNN kernels.

Other extension possibilities include testing other algorithms for the detection of anomalies in time-series data and playing with the analysis time frame. As already mentioned at the end of Section 2.4, there exists a wide range of possible MLAs based on different mathematical concepts. Additionally, by using longer time-series with longer intervals between measurements, it could be tested if MLAs can also recognize significant long term changes.

7.4 System Evaluation on Irregular Medical Data

The system has so far only been tested for the detection of various activities, which where defined in advance as artificial anomalies. In order to use it for the initially defined scope in the medical field, it must be tested more extensively using real irregular medical data. But, as mentioned in the introduction chapter, irregular medical data generation and recording is non-trivial. However, there are some possible cases.

A simpler one includes the measurements of irregular data during influenza illness. Every year there are seasonal influenza epidemics during the winter months [105]. This should make it easier to use the system to record measurements during fevers and other symptoms. Evaluating the implemented system and its MLAs on this data could show that this type of regularly occurring disease can be recognized as anomaly.

In another example, the system could be applied to cardiovascular diseases such as heart attacks or strokes, the most common causes of deaths worldwide [106]. The application of the system could be used to detect emerging irregularities, which are indicative of disease onset. It would open up the possibilities to apply appropriate treatments earlier and thus prevent possible deaths.

Bibliography

[1] ICT Facts and Figures, 2017. URL https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2017.pdf. [online; accessed 2018-06-15].

[2] Accelerating affordable smartphone ownership in emerging markets, 2017. URL https://www.gsma.com/mobilefordevelopment/wp-content/uploads/2017/07/accelerating-affordable-smartphone-ownership-emerging-markets-2017.pdf. [online; accessed 2018-06-15].

[3] CCS Insight Forecast Predicts Apple Watch and Hearables to Fuel Growth in Wearables, 2018. URL https://www.ccsinsight.com/press/company-news/3375-ccs-insight-forecast-predicts-apple-watch-and-hearables-to-fuel-growth-in-wearables. [online; accessed 2018-05-17].

[4] Kanitthika Kaewkannate and Soochan Kim. A comparison of wearable fitness devices. *BMC Public Health*, 16(1):433, 2016.

[5] Matthew J Bietz, Cinnamon S Bloss, Scout Calvert, Job G Godino, Judith Gregory, Michael P Claffey, Jerry Sheehan, and Kevin Patrick. Opportunities and challenges in the use of personal health data for health research. *Journal of the American Medical Informatics Association*, 23(e1):e42–e48, 2016.

[6] Choong Ho Lee and Hyung-Jin Yoon. Medical big data: promise and challenges. *Kidney research and clinical practice*, 36(1):3–11, 2017.

[7] Vital Signs (Body Temperature, Pulse Rate, Respiration Rate, Blood Pressure). URL https://www.hopkinsmedicine.org/healthlibrary/conditions/cardiovascular_diseases/vital_signs_body_temperature_pulse_rate_respiration_rate_blood_pressure_85,P00866. [online; accessed 2018-06-15].

[8] C. Bouchard and T. Rankinen. Individual differences in response to regular physical activity. *Medicine and science in sports and exercise*, 33(6 Suppl):446–51; discussion 452–3, 2001.

[9] Ziad Obermeyer, Jasmeet K. Samra, and Sendhil Mullainathan. Individual differences in normal body temperature: longitudinal big data analysis of patient records. *BMJ (Clinical research ed.)*, 359:j5468, 2017.

[10] Farah Shamout, David Clifton, and Tingting Zhu. Age- and Sex-Based Early Warning Score, 2017. URL `http://www.robots.ox.ac.uk/~davidc/pubs/transfer_fs.pdf`. [online; accessed 2018-06-15].

[11] Sue Sendelbach and Marjorie Funk. Alarm Fatigue. *AACN Advanced Critical Care*, 24(4):378–386, 2013.

[12] Mariska Weenk, Harry van Goor, Bas Frietman, Lucien J. Engelen, Cornelis J. van Laarhoven, Jan Smit, Sebastian J. Bredie, and Tom H. van de Belt. Continuous Monitoring of Vital Signs Using Wearable Devices on the General Ward: Pilot Study. *JMIR mHealth and uHealth*, 5(7):e91, 2017.

[13] Bertalan Mesko. The role of artificial intelligence in precision medicine. *Expert Review of Precision Medicine and Drug Development*, 2(5):239–241, 2017.

[14] Charu C. Aggarwal. Outlier Analysis. chapter 6, pages 169 – 198. Springer New York, 2013.

[15] Thomas Ahrens. The most important vital signs are not being measured. *Australian Critical Care*, 21(1):3–5, 2008.

[16] Aida Kamišalić, Iztok Fister, Muhamed Turkanović, Sašo Karakatič, and Sašo Karakatič. Sensors and Functionalities of Non-Invasive Wrist-Wearable Devices: A Review. *Sensors*, 18(6), 2018.

[17] Temperature Regulation and Fever. URL `http://m-learning.zju.edu.cn/G2S/eWebEditor/uploadfile/20111121083505_460699365083.pdf`. [online; accessed 2018-07-04].

[18] Greg Kelly. Body Temperature Variability (Part 1): A Review of the History of Body Temperature and its Variability Due to Site Selection, Biological Rhythms, Fitness, and Aging. *Altern Med Rev*, 11(4):278–293, 2006.

[19] Daniel Moran and Liran Mendal. Core temperature measurement: Methods and current insights. *Sports medicine*, 32(14):879–85, 2002.

[20] H.K. Walker, W.D. Hall, and J.W. Hurst. Temperature. In *Clinical Methods: The History, Physical, and Laboratory Examinations*, chapter 218. Butterworths, 3rd edition, 1990.

[21] Dennis L. Kasper, Anthony S. Fauci, Stephen L. Hauser, Dan L. Longo, J. Larry Jameson, and Joseph Loscalzo. Harrison's principles of internal medicine. McGraw-Hill Education, New York, 19th edition, 2015.

[22] Marie-Luise Blue. How Does the Body Regulate Heart Rate? URL `https://sciencing.com/body-regulate-heart-rate-19639.html`. [online; accessed: 2018-07-10].

[23] American Heart Association. All About Heart Rate (Pulse). URL `https://www.heart.org/HEARTORG/Conditions/HighBloodPressure/GettheFactsAboutHighBloodPressure/All-About-Heart-Rate-Pulse_UCM_438850_Article.jsp`. [online; accessed 2018-07-10].

[24] Obrey Alexis. Providing best practice in manual pulse measurement. *British Journal of Nursing*, 19(4):228–234, 2010.

[25] Dustin T. Weiler, Stefanie O. Villajuan, Laura Edkins, Sean Cleary, and Jason J. Saleem. Wearable Heart Rate Monitor Technology Accuracy in Research: A Comparative Study Between PPG and ECG Technology. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 61(1):1292–1296, 2017.

[26] Andreas Rieger. *Entwicklung und Konzeption eines Gehörgangsensors für die mobile Pulsoximetrie*. PhD thesis, Technische Universität München, 2010.

[27] Tachycardia - Symptoms and causes. URL `https://www.mayoclinic.org/diseases-conditions/tachycardia/symptoms-causes/syc-20355127`. [online; accessed 2018-07-10].

[28] Bradycardia - Symptoms and causes. URL `https://www.mayoclinic.org/diseases-conditions/bradycardia/symptoms-causes/syc-20355474`. [online; accessed 2018-07-10].

[29] Wiebke Sieben, Karen Schettlinger, Silvia Kuhls, Michael Imhoff, and Ursula Gather. Machine learning techniques in intensive care monitoring. 2008.

[30] M. C. Chambrin. Alarms in the intensive care unit: how can the number of false alarms be reduced? *Critical care*, 5(4):184–8, 2001.

[31] Michael Imhoff and Silvia Kuhls. Alarm Algorithms in Critical Care Monitoring. *Anesthesia & Analgesia*, 102(5):1525–1537, 2006.

[32] Fahmi Ben Rejab, Kaouther Nouira, and Abdelwahed Trabelsi. Health Monitoring Systems Using Machine Learning Techniques. In *Intelligent Systems for Science and Information*, pages 423–440. Springer, Cham, 2014.

[33] Lujie Chen, Artur Dubrawski, Donghan Wang, Madalina Fiterau, Mathieu Guillame-Bert, Eliezer Bose, Ata M. Kaynar, David J. Wallace, Jane Guttendorf, Gilles Clermont, Michael R. Pinsky, and Marilyn Hravnak. Using Supervised Machine Learning to Classify Real Alerts and Artifact in Online Multisignal Vital Sign Monitoring Data. *Critical Care Medicine*, 44(7):e456–e463, 2016.

[34] Ying Zhang. Real-Time Development of Patient-Specific Alarm Algorithms for Critical Care. In *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4351–4354. IEEE, 2007.

[35] M. Cardona-Morrell, M. Prgomet, R. M. Turner, M. Nicholson, and K. Hillman. Effectiveness of continuous or intermittent vital signs monitoring in preventing adverse events on general wards: a systematic review and meta-analysis. *International Journal of Clinical Practice*, 70(10):806–824, 2016.

[36] Tadashi Kamio, Tomoaki Van, and Ken Masamune. Use of Machine-Learning Approaches to Predict Clinical Deterioration in Critically Ill Patients: A Systematic Review. *International Journal of Medical Research & Health Sciences*, 6(6):1–7, 2017.

[37] Osman Salem, Alexey Guerassimov, Ahmed Mehaoua, Anthony Marcus, and Borko Furht. Anomaly Detection in Medical Wireless Sensor Networks using SVM and Linear Regression Models. *International Journal of E-Health and Medical Communications*, 5(1):20–45, 2014.

[38] Chelsea G. Bender, Jason C. Hoffstot, Brian T. Combs, Sara Hooshangi, and Justin Cappos. Measuring the fitness of fitness trackers. In *2017 IEEE Sensors Applications Symposium (SAS)*, pages 1–6. IEEE, 2017.

[39] Overviews of Activity and Fitness Trackers by Category. URL `http://www.bestfitnesstrackerreviews.com/overviews.html`. [online; accessed 2018-07-12].

[40] Peter Düking, Andreas Hotho, Hans-Christer Holmberg, Franz Konstantin Fuss, and Billy Sperlich. Comparison of Non-Invasive Individual Monitoring of the Training and Health of Athletes with Commercially Available Wearable Technologies. *Frontiers in physiology*, 7:71, 2016.

[41] Fatema El-Amrawy and Mohamed Ismail Nounou. Are Currently Available Wearable Devices for Activity Tracking and Heart Rate Monitoring Accurate, Precise, and Medically Beneficial? *Healthcare informatics research*, 21(4):315–20, 2015.

[42] Stephen Gillinov, Muhammad Etiwy, Robert Wang, Gordon Blackburn, Dermot Phelan, Marc Gillinov, Penny Houghtaling, Hoda Javadikasgari, and Milind Y. Desai. Variable Accuracy of Wearable Heart Rate Monitors during Aerobic Exercise. *Medicine and science in sports and exercise*, 49(8):1697–1703, 2017.

[43] Z. Ge, P. W. C. Prasad, N. Costadopoulos, Abeer Alsadoon, A. K. Singh, and A. Elchouemi. Evaluating the accuracy of wearable heart rate monitors. In *2016 2nd International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Fall)*, pages 1–6. IEEE, 2016.

[44] Sandra Schlichenmaier. Pulsmessung im Ohr: Der cosinuss One im Test. URL `https://knowledge.time2tri.me/tests/pulsmessung-im-ohr-der-cosinuss-one-im-test`. [online; accessed 2018-07-30].

[45] Testbericht: Der Cosinuss One - Klein, aber ohooo!, 2017. URL `https://tri-it-fit.de/testbericht-der-cosinuss-one-klein-aber-oho/`. [online; accessed 2018-07-30].

[46] Cosinuss One Puls-Ohrsensor. URL `https://www.mtb-news.de/forum/t/cosinuss-one-puls-ohrsensor.819239/`. [online; accessed 2018-07-30].

[47] George Coulouris, Jean Dollimore, Tim Kindberg, and Gordon Blair. Distributed systems : concepts and design. page 1047. Addison-Wesley, 2012.

[48] Sukumar. Ghosh. Distributed systems : an algorithmic approach. page 402. Chapman & Hall, 2007.

[49] Andrew S. Tanenbaum and Maarten van Steen. Distributed Systems: Principles and Paradigms. page 686. Pearson Prentice Hall, 2007.

[50] Specifications. URL `https://www.bluetooth.com/specifications`. [online; accessed 2018-07-06].

[51] New Bluetooth Standard Built for the Universal Remote Control. URL `https://www.bluetooth.com/news/pressreleases/2008/07/22/new-bluetoothstandard-built-for-the-universal-remote-control`. [online; accessed 2018-07-06].

[52] GATT Overview. URL `https://www.bluetooth.com/specifications/gatt/generic-attributes-overview`. [online; accessed 2018-07-06].

[53] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.

[54] Leonard Richardson and Michael Amundsen. RESTful Web APIs. O'Reilly, 2013.

[55] Overview - FHIR v3.0.1. URL `http://hl7.org/fhir/overview.html`. [online; accessed 2018-07-06].

[56] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78, 2012.

[57] Taiwo Oladipupo. Types of Machine Learning Algorithms. In *New Advances in Machine Learning*. InTech, 2010.

[58] Taiwo Oladipupo. Machine Learning Overview. In *New Advances in Machine Learning*. InTech, 2010.

[59] Qiujun Huang, Jingli Mao, and Yong Liu. An improved grid search algorithm of SVR parameters optimization. In *2012 IEEE 14th International Conference on Communication Technology*, pages 1022–1026. IEEE, 2012.

[60] Gian Antonio Susto, Angelo Cenedese, and Matteo Terzi. Time-Series Classification Methods: Review and Applications to Power Systems Data. In *Big Data Application in Power Systems*, chapter 9, pages 179–220. Elsevier, 2018.

[61] D. M. Hawkins. Identification of Outliers. Springer Netherlands, 1980.

[62] Victoria J Hodge and Jim Austin. A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review Artificial Intelligence ReviewJ. and Austin, J. Artificial Intelligence Review*, 22(222):85–126, 2004.

[63] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly Detection: A Survey. *ACM Computing Surveys*, 41(3):1–58, 2009.

[64] H. Kriegel, P. Kroger, and A. Zimek. Outlier Detection Techniques. *The 2010 SIAM International Conference on Data Mining*, 2010. URL `https://archive.siam.org/meetings/sdm10/tutorial3.pdf`. [online; accessed 2018-06-28].

[65] Poonam Rana, Deepika Pahuja, and Ritu Gautam. A Critical Review on Outlier Detection Techniques. *International Journal of Science and Research*, 3(12):2319–7064, 2012.

[66] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Outlier Detection: A Survey. Technical report, University of Minnesota, 2007.

[67] How to choose machine learning algorithms. URL `https://docs.microsoft.com/en-us/azure/machine-learning/studio/algorithm-choice`. [online; accessed 2018-07-04].

[68] Which machine learning algorithm should I use? URL `https://blogs.sas.com/content/subconsciousmusings/2017/04/12/machine-learning-algorithm-use/`. [online; accessed 2018-07-04].

[69] Choosing the right estimator. URL `http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html`. [online; accessed 2018-07-04].

[70] S. S. Sreevidya. A Survey on Outlier Detection Methods. *International Journal of Computer Science and Information Technologies (IJCSIT)*, 5(6), 2014.

[71] Marco A. F. Pimentel, David A. Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[72] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, Jörg Sander, Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: Identifying Density-Based Local Outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data - SIGMOD '00*, volume 29, pages 93–104. ACM Press, 2000.

[73] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation Forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[74] J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986.

[75] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[76] K.-R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2): 181–201, 2001.

[77] Hyun Joon Shin, Dong-Hwan Eom, and Sung-Shick Kim. One-class support vector machines—an application in machine fault detection and classification. *Computers & Industrial Engineering*, 48(2):395–408, 2005.

[78] Introduction to One-Class Support Vector Machines. URL `http://rvlasveld.github.io/blog/2013/07/12/introduction-to-one-class-support-vector-machines/`. [online; accessed 2018-07-04].

[79] Djeffal Abdelhamid, Mohamed Mounir Souissi, and Abdelhey Chelouai. 3d support vector clustering for fingerprints segmentation. *International Conference on Knowledge Discovery and Data Analysis*, pages 119–131, 2015.

[80] David E. Rumelhart, James L. McClelland, and San Diego. PDP Research Group. University of California. Parallel distributed processing : explorations in the microstructure of cognition. In *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1*, pages 318–362. MIT Press, 1986.

[81] Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58, 1989.

[82] S. Agatonovic-Kustrin and R. Beresford. Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. *Journal of Pharmaceutical and Biomedical Analysis*, 22(5):717–727, 2000.

[83] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

[84] D. O. Hebb. The organization of behavior: A neuropsychological theory. *Science Education*, 34(5):336–337, 1950.

[85] Understanding Activation Functions in Neural Networks. URL `https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0`. [online; accessed 2018-07-05].

[86] Loss Functions in Neural Networks, 2017. URL `https://isaacchanghau.github.io/post/loss_functions/`. [online; accessed 2018-07-05].

[87] The mostly complete chart of Neural. URL `https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464`. [online; accessed 2018-07-05].

[88] Marco Schreyer, Timur Sattarov, Damian Borth, Andreas Dengel, and Bernd Reimer. Detection of Anomalies in Large Scale Accounting Data using Deep Autoencoder Networks. *ArXiv*, 2017.

[89] IDC: Smartphone OS Market Share. URL `https://www.idc.com/promo/smartphone-market-share/os`. [online; accessed 2018-08-06].

[90] One - Cosinuss, 2018. URL `https://www.cosinuss.com/one/`. [online; accessed 2018-07-12].

[91] Cosinuss One Technisches Datenblatt. URL `https://www2.cosinuss.com/content/7-support/datenblatt_cosinuss-one_v1.1-25072016-gs-deutsch.pdf`. [online; accessed 2018-07-12].

[92] James Agnew. HAPI FHIR - The Open Source FHIR API for Java. URL `http://hapifhir.io/`. [online; accessed 2018-08-03].

[93] Inc. Square. Retrofit: A type-safe HTTP client for Android and Java, 2013. URL `https://square.github.io/retrofit/`. [online; accessed 2018-08-03].

[94] Django Software Foundation and Individual Contributors. Django - The Web framework for perfectionists with deadlines, 2005. URL `https://www.djangoproject.com/`. [online; accessed 2018-08-03].

[95] Tom Christie. Django REST framework. URL `http://www.django-rest-framework.org/`. [online; accessed 2018-08-03].

[96] Josh Mandel, Nikolai Schwertner, and Pascal Pfiffner. SMART on FHIR - Clients - Python. URL `http://docs.smarthealthit.org/clients/python/`. [online; accessed 2018-08-03].

[97] Travis Oliphant. Guide to numpy, 2006. URL `http://web.mit.edu/dvp/Public/numpybook.pdf`. [online; accessed 2018-08-03].

[98] Wes McKinney. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[99] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[100] François Chollet. Keras, 2015. URL `https://keras.io/`. [online; accessed 2018-08-03].

[101] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. [online; accessed 2018-08-03].

[102] John D. Hunter. Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[103] Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W. Kempa-Liehr. Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package). *Neurocomputing*, 307:72–77, 2018.

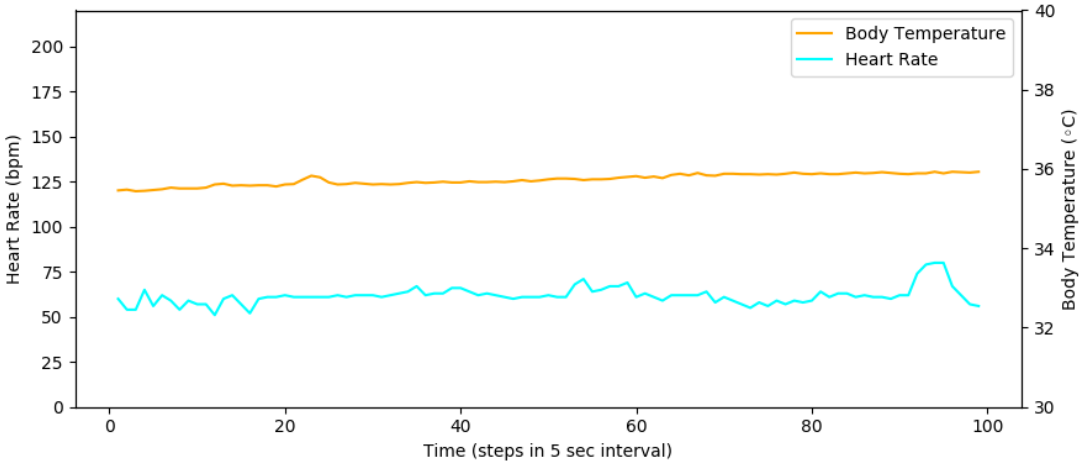
[104] A. Mahabal, K. Sheth, F. Gieseke, A. Pai, S. G. Djorgovski, A. J. Drake, and M. J. Graham. Deep-learnt classification of light curves. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8. IEEE, 2017.

[105] World Health Organization. Influenza (Seasonal). URL `http://www.who.int/en/news-room/fact-sheets/detail/influenza-(seasonal)`. [online; accessed 2018-08-06].

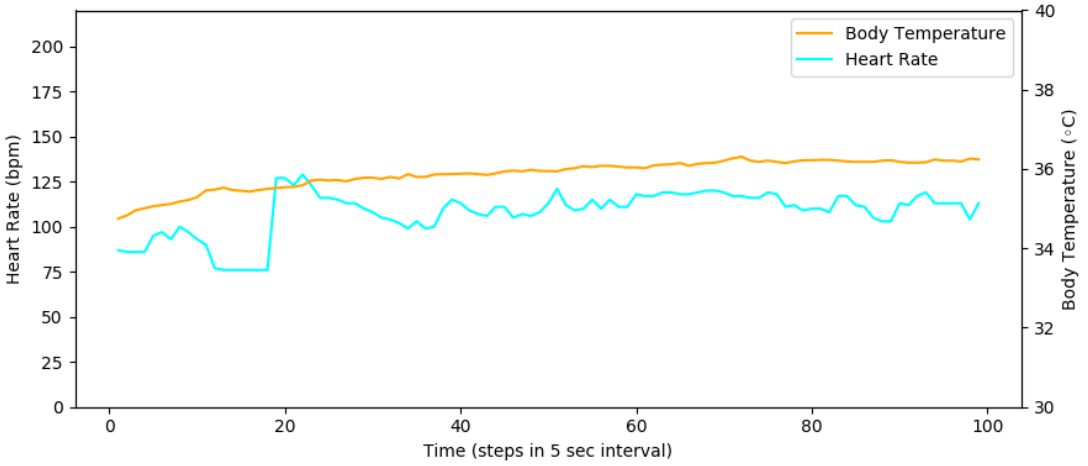
[106] World Health Organization. Hearts: Technical package for cardiovascular disease management in primary health care. Technical report, 2016.

Appendix A

Visualized Measurements

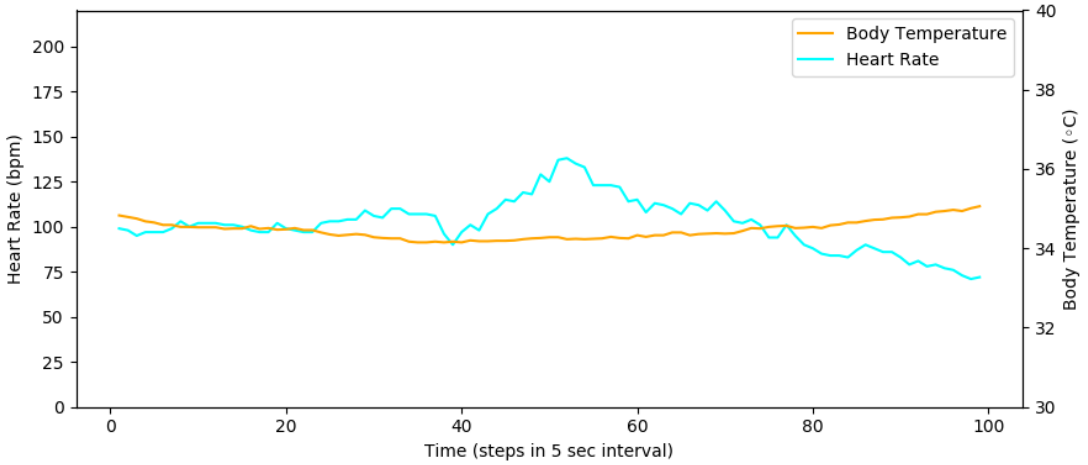


(a)

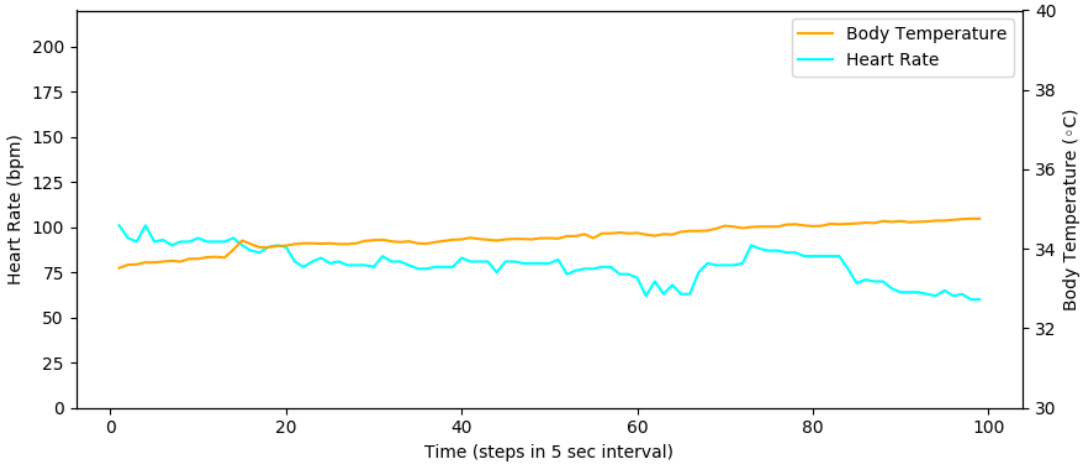


(b)

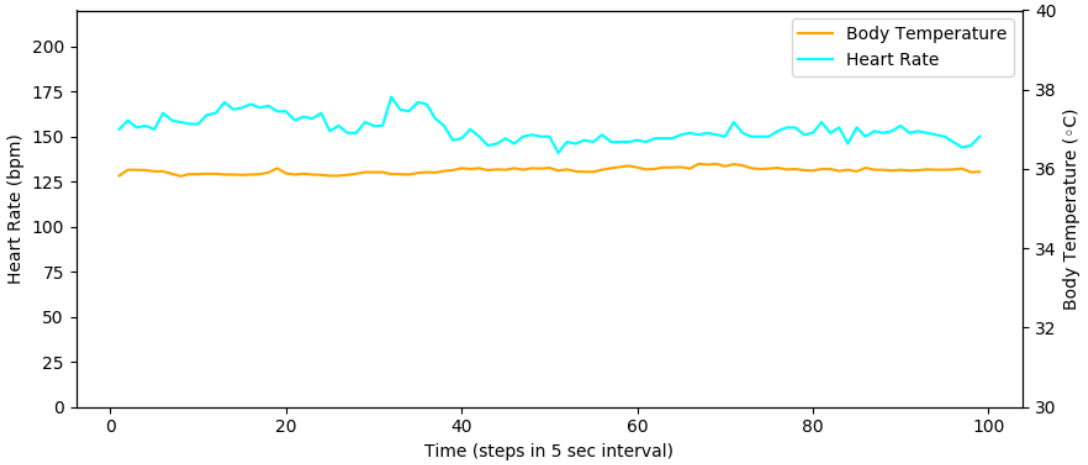
FIGURE A.1: Comparison of body temperature and heart rate measurements between the activities sitting and eating: (a) sitting, (b) eating.



(a)



(b)



(c)

FIGURE A.2: Comparison of body temperature and heart rate measurements between the activities walking, metro and sport: (a) walking, (b) metro, (c) sport.

Appendix B

FHIR JSON Messages

```
{
   "resourceType":"Observation",
   "status":"final",
   "category":[
      {
         "coding":[
            {
               "system":"http://hl7.org/fhir/observation-category",
               "code":"vital-signs",
               "display":"Vital Signs"
            }
         ],
         "text":"Vital Signs"
      }
   ],
   "code":{
      "coding":[
         {
            "system":"http://loinc.org",
            "code":"8310-5",
            "display":"Body temperature"
         },
         {
            "system":"http://snomed.info/sct",
            "code":"386725007",
            "display":"Body temperature (observable entity)"
         }
      ],
      "text":"Body temperature"
   },
   "subject":{
      "reference":"Patient/58aa5be2-3fe5-3e79-925a-3a626e6722a7",
      "display":"Anonymous Anonymous"
   },
   "effectiveDateTime":"2018-07-25T16:29:08-04:00",
   "interpretation":{
      "coding":[
```

```
38          {
39             "code":"00002",
40             "display":"Sitting",
41             "system":"http://fhir.mt.retwet.com/current-activity"
42          },
43          {
44             "code":"00001",
45             "display":"Inside",
46             "system":"http://fhir.mt.retwet.com/current-location"
47          }
48       ]
49    },
50    "issued":"2018-07-25T16:29:08.609-04:00",
51    "performer":[
52       {
53          "reference":"Patient/58aa5be2-3fe5-3e79-925a-3a626e6722a7",
54          "display":"Anonymous Anonymous"
55       }
56    ],
57    "valueQuantity":{
58       "value":35.259998321533203125,
59       "unit":"C",
60       "system":"http://unitsofmeasure.org",
61       "code":"Cel"
62    },
63    "bodySite":{
64       "coding":[
65          {
66             "system":"http://snomed.info/sct",
67             "code":"25342003",
68             "display":"Middle ear structure (body structure)"
69          }
70       ],
71       "text":"Middle ear structure (body structure)"
72    },
73    "method":{
74       "coding":[
75          {
76             "system":"http://snomed.info/sct",
77             "code":"56342008",
78             "display":"Temperature taking (procedure)"
79          }
80       ],
81       "text":"Temperature taking (procedure)"
82    }
83 }
```

Listing B.1: Example of a labeled body temperature measurement encoded as Health Level 7 Fast Healthcare Interoperable Resources Observation (STU3) using JSON format.

```
1  {
2     "resourceType":"Observation",
3     "status":"final",
```

```
   "category":[
      {
         "coding":[
            {
               "system":"http://hl7.org/fhir/observation-category",
               "code":"vital-signs",
               "display":"Vital Signs"
            }
         ],
         "text":"Vital Signs"
      }
   ],
   "code":{
      "coding":[
         {
            "system":"http://loinc.org",
            "code":"8867-4",
            "display":"Heart rate"
         },
         {
            "system":"http://snomed.info/sct",
            "code":"364075005",
            "display":"Heart rate (observable entity)"
         }
      ],
      "text":"Heart rate"
   },
   "subject":{
      "reference":"Patient/58aa5be2-3fe5-3e79-925a-3a626e6722a7",
      "display":"Anonymous Anonymous"
   },
   "effectiveDateTime":"2018-07-25T16:29:08-04:00",
   "issued":"2018-07-25T16:29:08.605-04:00",
   "performer":[
      {
         "reference":"Patient/58aa5be2-3fe5-3e79-925a-3a626e6722a7",
         "display":"Anonymous Anonymous"
      }
   ],
   "valueQuantity":{
      "value":61,
      "unit":"beats/minute",
      "system":"http://unitsofmeasure.org",
      "code":"/min"
   },
   "bodySite":{
      "coding":[
         {
            "system":"http://snomed.info/sct",
            "code":"25342003",
            "display":"Middle ear structure (body structure)"
         }
      ],
      "text":"Middle ear structure (body structure)"
   },
```

```
59      "method":{
60         "coding":[
61            {
62               "system":"http://snomed.info/sct",
63               "code":"72075005",
64               "display":"Photoplethysmography (procedure)"
65            }
66         ],
67         "text":"Photoplethysmography (procedure)"
68      }
69   }
```

LISTING B.2: Example of an unlabeled heart rate measurement encoded as Health Level 7 Fast Healthcare Interoperable Resources Observation (STU3) using JSON format.

```
1   {
2      "category":[
3         {
4            "coding":[
5               {
6                  "code":"procedure",
7                  "display":"Procedure",
8                  "system":"http://hl7.org/fhir/observation-category"
9               }
10           ],
11           "text":"Procedure"
12        }
13     ],
14     "code":{
15        "coding":[
16           {
17              "code":"anomaly-classification",
18              "display":"Anomaly Classification",
19              "system":"http://fhir.mt.retwet.com/vital-sign-classification"
20           }
21        ],
22        "text":"Anomaly Classification"
23     },
24     "effectiveDateTime":"2018-07-27T15:41:00.788376-04:00",
25     "interpretation":{
26        "coding":[
27           {
28              "code":"N",
29              "display":"Normal",
30              "system":"http://hl7.org/fhir/v2/0078"
31           }
32        ],
33        "text":"Normal"
34     },
35     "issued":"2018-07-27T15:41:00.788474-04:00",
36     "method":{
37        "coding":[
38           {
```

```
39              "code":"isolation-forest",
40              "display":"Isolation Forest",
41              "system":"http://fhir.mt.retwet.com/machine-learning-classification"
42           }
43        ],
44        "text":"Isolation Forest"
45     },
46     "performer":[
47        {
48           "display":"Master Thesis Server",
49           "reference":"Organization/12v43g60-bj3f-34j5-db53-867dtgff8rmsa"
50        }
51     ],
52     "status":"final",
53     "subject":{
54        "display":"Anonymous Anonymous",
55        "reference":"Patient/patient-58aa5be2-3fe5-3e79-925a-3a626e6722a7"
56     },
57     "resourceType":"Observation"
58  }
```

Listing B.3: Example of a classification result encoded as Health Level 7 Fast Healthcare Interoperable Resources Observation (STU3) using JSON format.

Appendix C

Android Application Screenshots

The following eight figures show screenshots of the different Android application views, in order of their use:

- **Figure C.1:** The initialize view to set the initial settings needed to run the application.
- **Figure C.2:** The about view showing information about the Android application and why it was developed.
- **Figure C.3:** The main view before the connection to the Cosinuss° One sensor is established.
- **Figure C.4:** The connection view during search and establishment of the connection to the Cosinuss° One sensor.
- **Figure C.5:** The main view displaying the BT and HR measurements received from the Cosinuss° One sensor and the dropdown fields for labeling the training data.
- **Figure C.6:** The main view displaying the classification result received from the Django ML server.
- **Figure C.7:** The settings view showing the different options.
- **Figure C.8:** The subsettings view for selection of a MLA.

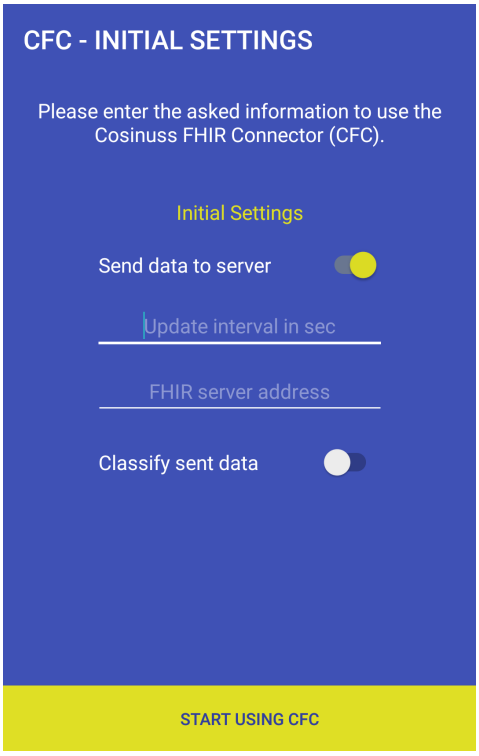


FIGURE C.1: Screenshot initialization.

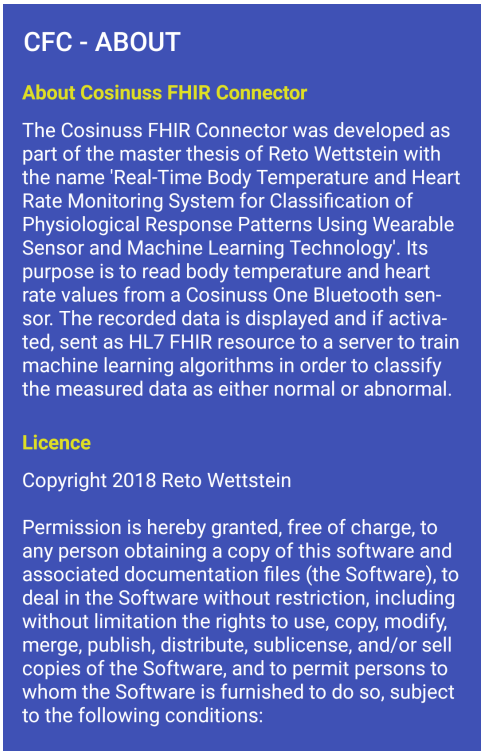


FIGURE C.2: Screenshot about view.

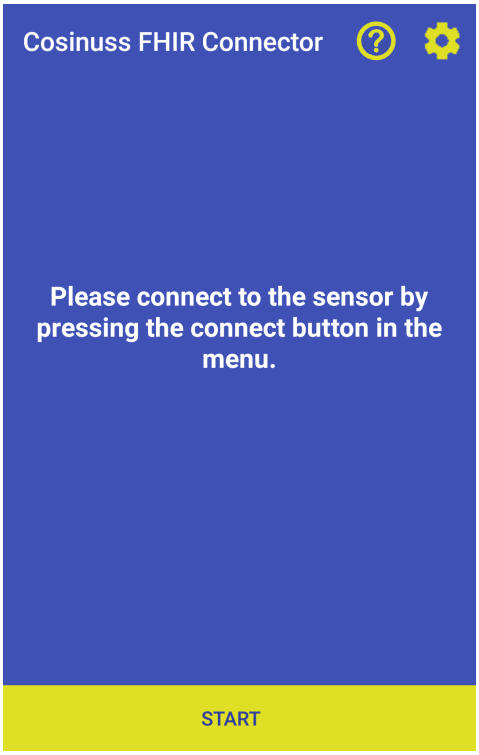


FIGURE C.3: Screenshot before connection establishment.

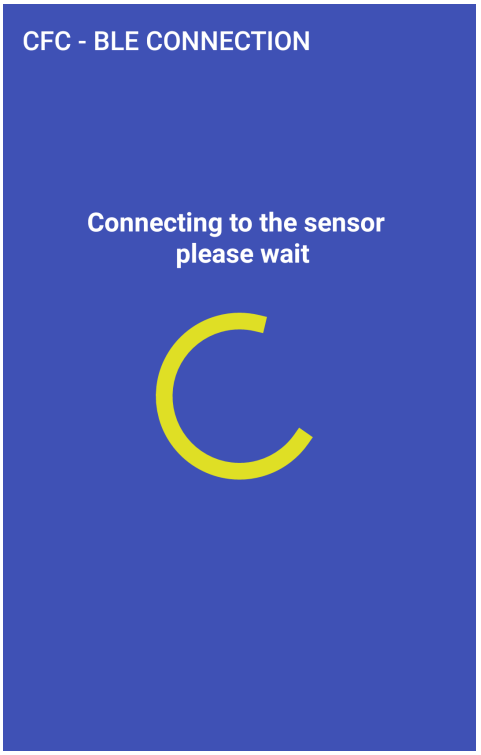


FIGURE C.4: Screenshot during connection establishment.

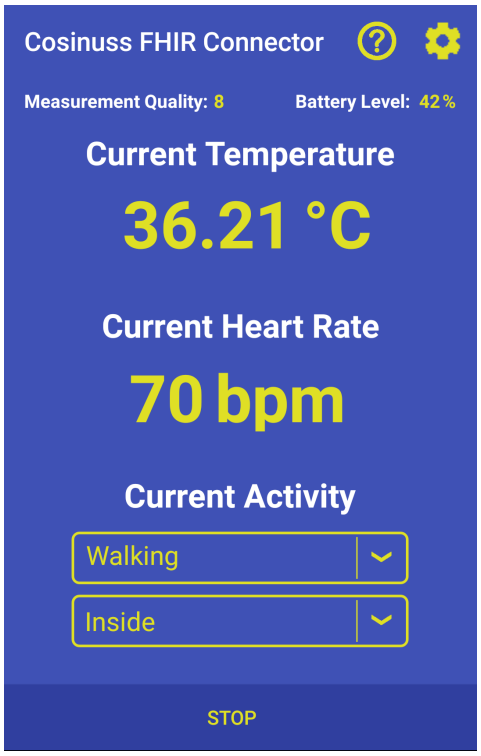


FIGURE C.5: Screenshot training.

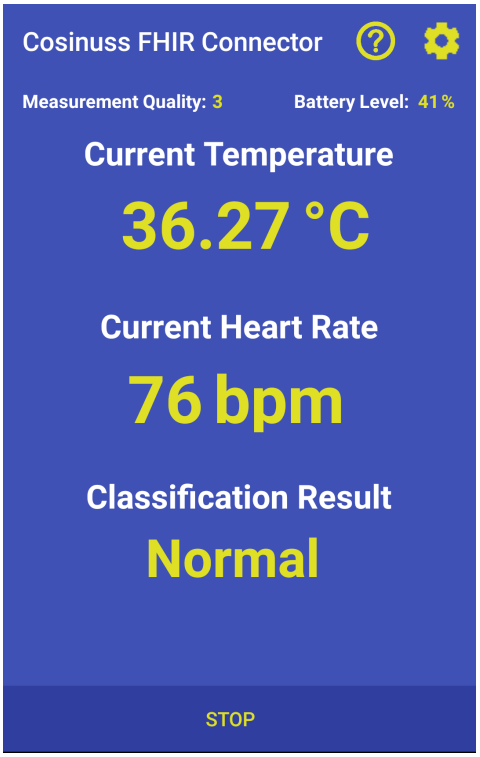


FIGURE C.6: Screenshot classification.

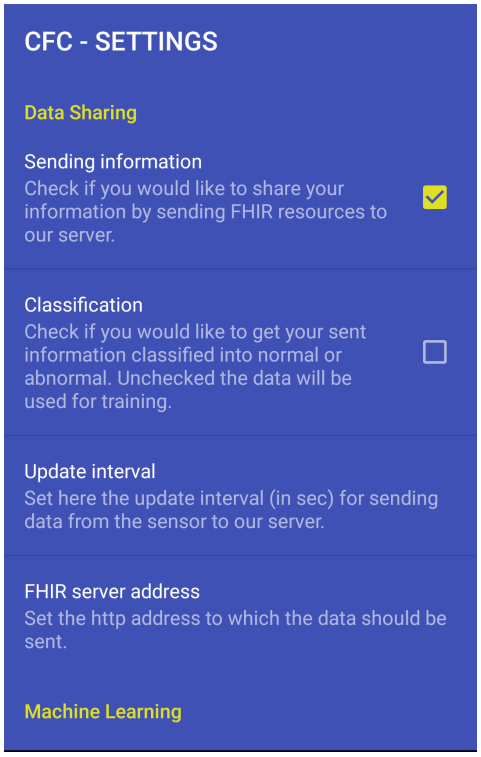


FIGURE C.7: Screenshot overall settings.

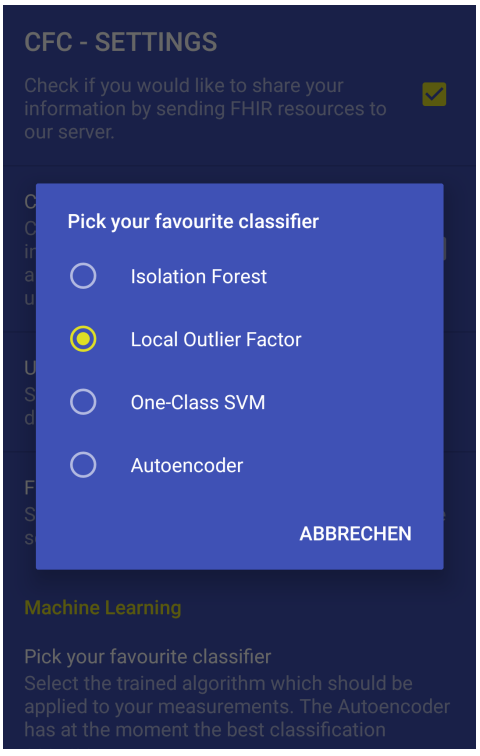


FIGURE C.8: Screenshot settings for machine learning algorithm selection.

Appendix D

Digital Contents

The CD attached to this appendix has the following contents:

- Complete thesis in the file *thesis.pdf* and the corresponding presentation in the file *presentation.pdf*.
- Java code of the implemented Android application in the folder *client*.
- Python code of the implemented Django ML server in the folder *server*.
- Recorded measurements during the experiment to test the proof-of-concept prototype in the folder *server/files/storage*.