Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Andreas Kamenz

Time Series Analysis of Physiological Data for Emotion Recognition using Deep Learning

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Andreas Kamenz

Time Series Analysis of Physiological Data for Emotion Recognition using Deep Learning

Masterarbeit eingereicht im Rahmen der Masterprüfung

im Studiengang Master of Science Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Kai von Luck; Prof. Dr. Tim Tiedemann

Eingereicht am: 27. Januar 2019

Andreas Kamenz

Thema der Arbeit

Zeitreihenanalyse von physiologischen Sensordaten zur Emotionserkennung mittels Deep Learning

Stichworte

Deep Learning, Machine Learning, Companion Technology, Affective Computing, Emotion Recognition, Time Series Classification, Body Monitoring

Kurzzusammenfassung

Eine gute Emotionserkennung ist für Companion Systeme von großer Bedeutung, um bestmöglich auf den Benutzer antworten zu können. In dieser Masterarbeit wird der erstellte Versuchsaufbau und der durchgeführte Laborversuch im Kontext Emotionserkennung beschrieben. Mehrere Machine- und Deep-Learning Verfahren werden vorgeschlagen, um dieses Problem zu lösen. Diese werden implementiert und miteinander verglichen, um einen Beitrag zu besseren Ergebnissen bei der Emotionserkennung zu leisten.

Andreas Kamenz

Title of the paper

Time Series Analysis of Physiological Data for Emotion Recognition using Deep Learning

Keywords

Deep Learning, Machine Learning, Companion Technology, Affective Computing, Emotion Recognition, Time Series Classification, Body Monitoring

Abstract

A good emotion recognition is of great importance for companion systems in order to be able to respond to the user in the best possible way. In this master thesis the test setup and the laboratory study will be described. We propose serveral machine and deep learning methods to solve this problem. These methods will be implemented and compared to show that they contribute to an improved emotion recognition performance.

Acknowledgements

At this point, I would like to start with Prof. Dr. Kai v. Luck and Prof. Dr. Tim Tiedemann, thank you very much for the support of this work.
Special thanks goes also to my wife Victoria Bibaeva for the extensive support and motivation. And finally, last but by no means least, also to everyone from the EmotionBike project, especially Larissa, Arne and Sobin. It was a great sharing laboratory with all of you during the last four years, with many long days at the numerous hackathons and even longer nights in the Living Place.

Thanks for all your encouragement!

Contents

1 Introduction

1.1 Motivation

Intelligent systems – also called companion systems – are gaining more and more importance in our daily life. They can help people to cope with everyday life – for example, working as language assistants, such as Apple Siri[1], Google Assistant[2], Amazon Alexa[3], and Microsoft Cortana[4], or as robots like Pepper[5] and Buddy[6].

To enable these systems to react adequately to people, it is advantageous to interpret and derive human intentions. However, the recognition of emotions is a difficult task for companion systems. In research, there are different approaches to the recognition of emotions based on audio, video, or physiological data. Additionally, the above-mentioned data sources can be combined using multimodal methods. In contrast, audio- and video-based methods are much more deeply researched than methods based on physiological data, but they are limited in principle to linguistic utterances or facial emotions.

The scientific field for this systems is affective computing [PIC1997], which is an interdisciplinary field between computer science, neuroscience, physiology, psychology, sociology, and cognitive science. It includes topics like the following:

- Theoretical basics about the nature of motivation, emotions, and feelings
- The physiology of the brain and other aspects of human physiology relevant to affective states
- The relationship between emotions and intelligence
- Effective human-computer interfaces (HCI) and their requirements and realization

[1] https://www.apple.com/siri/
[2] https://assistant.google.com/
[3] https://developer.amazon.com/alexa/
[4] https://www.microsoft.com/en-us/cortana
[5] https://www.ald.softbankrobotics.com/en/robots/pepper
[6] http://www.bluefrogrobotics.com/en/buddy/

- Wearables with a wide range of sensing and communication possibilities
- Recognition of emotional and other affective states
- Make machines not only intelligent, but also empathic
- Philosophical and ethical issues relating to artificial intelligence and robotics

In this thesis, the focus is on "*recognition of emotional and other affective states.*"

The physiological data of a person (heart rate, blood pressure, electrical conductivity of the skin, body temperature, etc.) have some advantages to emotion recognition. However, they are difficult to fake, and they do not differ between different ethnic groups or cultures, but only from individual to individual.

The fields of application for such intelligent technical systems are manifold. Ultimately, their purpose is to enable an individually tailored reaction to the person or to provide suitable recommendations for action, whether in computer games, in smart homes, in fitness and health applications, or as assistance in everyday life.

Due to the rapid progress in the field of miniaturization of technical systems – like medical sensor technology and increasingly powerful hardware in general – such devices become cheaper and thus also affordable outside the health-care industry.

The Quantify Self community [7] has made a major contribution to this end. Through the will to collect it's own body data and through a variety of do-it-yourself kits, it can be done with the simplest means and for little money. At the same time, the processing of large amounts of data – such as sensor data – with machine and deep learning is getting easier and easier: first, through ever more sophisticated algorithms and scientific experiences in the scope of neural networks, and also through ever cheaper, and thereby increasing, computing capacities from a university cluster or cloud providers.

For this reason various possibilities have arisen for emotion recognition with physiological sensor data that did not exist a few years ago.

[7] https://quantifiedself.com/

1.2 Aim of the work

This work is intended to evaluate the time series data carried out within a laboratory study with 25 participants in the context of the EmotionBike research project at the Hamburg University of Applied Sciences (HAW Hamburg). For this purpose, various algorithms from the field of machine learning and deep learning were examined and compared. We used analytical methods that enable an automated pattern recognition and classification for a discrete emotional model with six basic emotions (anger, disgust, fear, joy, sadness, and surprise), according to Ekman [EFE1972].

1.3 Structure of the work

This master thesis is divided into six chapters. The first chapter addresses the motivation and objectives of the master thesis. In the second chapter, the experimental setup of the Emotion Bike research project is explained, including which hard- and software components were used to collect the physiological data. In the third chapter, the results from the literature review and the related work are outlined, as well as the placement of this thesis in the scientific research area. The proposed algorithms from the context of machine and deep learning for time series analysis are described in Chapter 4. The evaluation of the experimental results for automated emotion recognition are discussed in Chapter 5. Finally in Chapter 6, a conclusion is shown and an outlook on further goals is given.

2 EmotionBike - Research Project

2.1 System Setup

The experiments in this thesis were carried out within the context of the *EmotionBike* [1] research project at Hamburg University of Applied Sciences (HAW Hamburg). The EmotionBike is a laboratory environment with the aim of investigating the interaction of humans with computers (HCI) and the possibilities of companion technology in the context of exergames. Whereby *exergame* is a word composed of exercise (to train in the real world) and game (seeing your pedaling and steering movements in a virtual reality). The experimental setup consists of a modified ergometer, a flat screen display, and various sensors to record the current condition of a participant.

The base component of the EmotionBike environment is a bicycle ergometer tuned with a 180° flat handlebar [HOR2015], a brake [MAT2015], and software controlled pedal resistance to simulate the gradient and slope of a landscape. These components enables a realistic control of the bicycle in the virtual world, therefore achieving a high immersion of the participant. An ergometer has the advantage in that the test subject is fixed in place. As a result there are fewer signal disturbances that can be triggered by rapid movements, and the participant is more focused on what is happening on the screen.

The second component is an emotion provoking game [ZAG2017] that is shown in front of the participant on a flat screen. The ergometer is the input device for controlling a bicycle in the virtual environment. There are several levels with distinct situations to provoke certain emotions. Each level begins and ends in a tunnel. This tunnel helps the participants to calm down between the levels and to bring the sensor data back to a baseline. Within a level, the participants are guided by the level structure exactly to certain areas, where an event is triggered, and as a result, the emotion can be recorded. All levels have been designed so that they are not overly strenuous for the participant, preventing the values from being dependent on the condition.

To make the results comparable, the participants are influenced by certain events at fixed points in the different levels and placed into an expected emotional state. This is done by the sudden

[1] http://emotionbike.org/

Figure 2.1: EmotionBike environment

occurrence of shock moments or difficult challenges, such as skipping an abyss. After each level, the respondent completes an additional questionnaire to capture a subjective impression of an emotional state for later evaluation.

A component diagram is shown in Appendix 6.1; the biosensor component is highlighted red.

2.1.1 Distributed System Architecture

The EmotionBike system uses a publish-subscribe architecture implemented by ActiveMQ [2] for communication between the individual nodes. There are topics for control commands to the nodes (EMOBIKE.CONTROL) and status messages from the nodes (EMOBIKE.STATUS).

[2] https://activemq.apache.org/

Via the topic *EMOBIKE.CONTROL* the control center component transmits the following control commands to the nodes:

sessionInfo	General information about the current test run (test ID, session ID, measure ID, storage path for the measurement data)
initializeMeasure	Prepare for measurement (initialize connection with Plux Hub)
startMeasure	Start measurement
stopMeasure	Stop measurement
doReset	Abort all currently running processes in a controlled manner and return to the initial state
getTime	Get current timestamp of a node

Table 2.1: Control commands

The second topic *EMOBIKE.STATUS* is used for status messages from the nodes. The following statuses are possible:

start	Node is booting
online	Node is reachable
ready	Node is ready for control commands
measurePrepared	Node is ready for measurement
working	Node is record an active measurement
writing	Node is writing the data to file
finished	Measurement completed and data stored
error	A runtime error has occurred
reset	Reset is performed

Table 2.2: Status messages

Additionally, there is a topic, *EMOBIKE.HEARTBEAT*, where all active nodes periodically send a message containing the current time stamp of the node every second. This message serves to recognize whether nodes failed without an error message. In the case of system-relevant nodes, the attempt can also be restarted in order to maintain data consistency for the attempt run.

Furthermore, the heartbeat serves to detect time asynchronism between the different nodes, which is a problem not to be underestimated in a distributed system due to network latencies and different time determination methods between the operating system and programming language.

The necessity of such measures for time synchronization had already arose during the evaluation of the first test measurements. The use of different accuracies (nanoseconds or microseconds) and undefined time zones (UTC, CET, or CEST) led to confusion regarding data quality. In

the later course of the regular experiments, there were still deviations in the millisecond to single-digit second range, but these were synchronized using heartbeat messages.

An overview of the node state machine is shown in Appendix 6.1.

2.2 Sensor System

2.2.1 Requirements

As noted in the basic [KAM2016] and main project [KAM2017], the sensor system must meet specific requirements:

- High accuracy of the recorded values at 256 Hz
- All necessary sensors included (ECG, BVP, RESP, TEMP, EDA, ACC)
- Unobtrusive sensors and wires (in order to not disturb/irritate the participants)
- Easy row data access for data mining
- Real-time visualization of the sensor data

2.2.2 Hardware

For the purpose of emotion recognition using physiological data, the Plux Hub[3] was integrated into the EmotionBike environment. This product was selected after various similar products were compared in preliminary studies, and this has asserted itself as best fitting. It is a toolkit for biomedical research to collect and evaluate physiological data. It consists of a central hub and a lot of different sensors. The hub can communicate to other computers via a wireless Bluetooth connection and the sensors are connected via USB wires to the hub. Five of the offered biosensors were selected and attached to the participant (EDA, ECG, BVP, respiration and temperature). In the following, the purpose and positioning of the sensors will be described in more detail.

EDA

With an EDA sensor the conductivity of the skin can be determined. Two electrodes are attached to the inside of the forefinger and middle finger (see Fig. 2.4). Due to perspiration on the skin, the skin conduction resistance changes and as a result the skin conduction capacity increases. This value is recorded by an the EDA sensor.

[3] http://biosignalsplux.com/en/

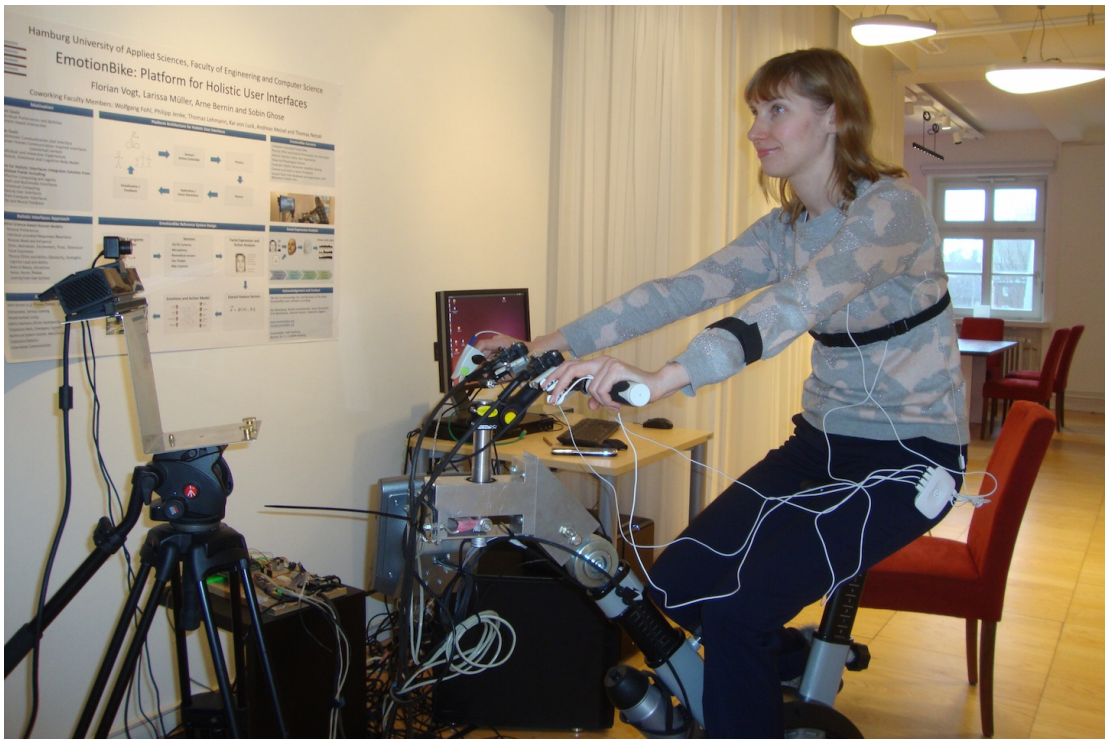


Figure 2.2: Participant with attached biosensors

ECG

The ECG sensor can be used to record an electrocardiogram, which is the cardiac tension curve of the individual. This record is captured by three electrodes which are attached to the surface of the skin; they detect potential electrical changes in the heart. The arrangement can take place in different patterns, depending on the number of electrodes and processes. We have opted for the Einthoven triangle placement [MIE2003] (see Fig. 2.3) as it provided the best results in preliminary tests.

With the Plux System, there are three electrodes at certain locations near the heart, and we use the three vertical axes to determine the heartbeat. Doing so allows fully accurate data on heart rhythm and heart rate to be recorded.

BVP

The BVG sensor is placed on the index finger and is used to collect data from the cardiovascular system. It works with infrared light by shining it onto the skin and determining the strength of its reflection. Blood in the veins leads to a stronger light absorption, and thus with each heartbeat at

Figure 2.3: A: ECG ground electrode, B: ECG anode, C: ECG cathode, D: Respiration belt, E: Temperature sensor

which blood is pumped into the veins, the heart rate can be determined. It is a simpler alternative to the ECG for heart rate measurement.

Respiration

A chest belt with piezo elements was used to record the respiration of a participant. The piezo effect is used for this purpose, whereby electrical charges are generated on the crystal surface of the piezo elements when they are mechanically deformed. This charge increases proportionally to the pressure exerted on the crystals. So, respiration frequency, rhythm, and intensity can be measured.

Temperature

The temperature sensor can be used to measure the body temperature of the participant. It is placed at the inner side of the left forearm between the ulnar and radial arteries (see Fig 2.4).

Acceleration

An three-axis accelerometer was placed at the left forearm to determine fast movements of the participant's arms. It is an indication that the recorded data from the other sensors are susceptible to interference at this point in time and may need to be post-processed in order to remove interferences.

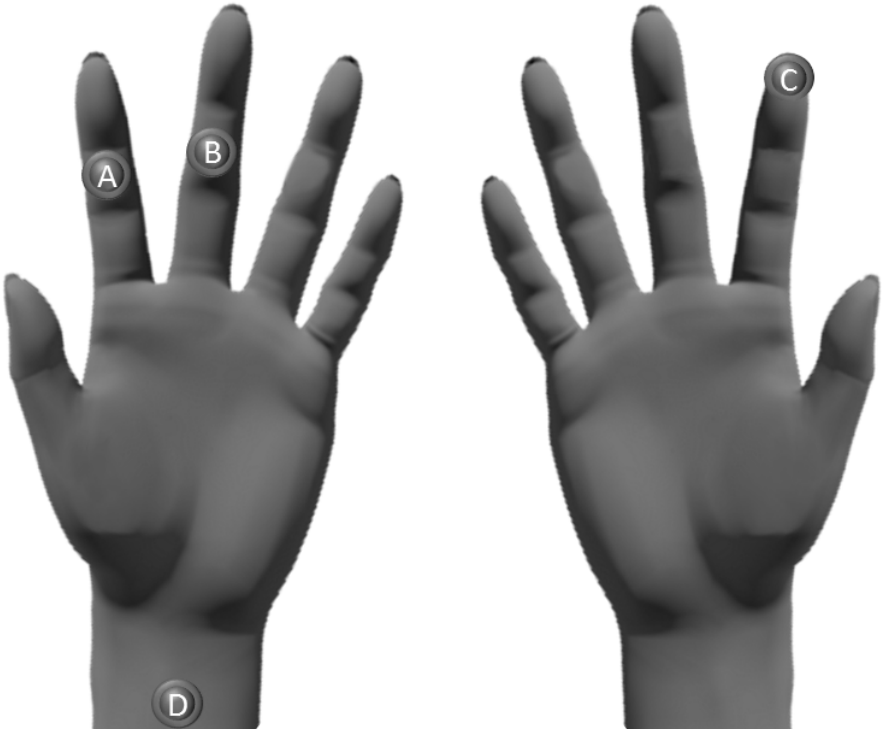


Figure 2.4: A: 1. EDA electrode, B: 2. EDA electrode, C: BVG sensor, D: 3-axis accelerometer

2.2.3 Software

A software consisting of several components was written in Python, see Appendix 6.1. This program was integrated as a node into the distributed system of the entire experimental setup and is used to control and collect data from the Plux hub.

The software consists of a total of five components: Sensor, Plotter, Processing, Communication, and Persistence (UML diagrams see Appendix 6.1).

Sensor component

The sensor component serves as an interface to the Plux hub. It consists of a *DataReceiver* class that implements various callback methods. These callbacks are called by the Plux hub when certain events occur. One of these callbacks is the *onRawFrame* method, in which the captured sensor data – including timestamp and sequence number – are periodically inserted into a queue for further processing. The second class in the sensor component is the *SensorService*, which is used to control the plux hub. The service establishes the Bluetooth connection, ensures error handling in case of connection problems, and starts or ends a measurement on the Plux hub.

Plotter component

A plotter component was developed to allow visual control during the experiment. It displays in real time the received sensor data as diagrams (example see Fig. 6.1) and optionally inserts the triggered events as vertical markers.

Processing component

Additionally, the processing component can be used to perform certain data analyses on a preconfigured time window and display the results in the diagram. For example, in the diagram, the peaks can be marked, or a moving average can be displayed.

However, restrictions on the algorithms had to be specified to ensure real-time capability. So far, only algorithms with low complexity have been implemented. Furthermore, not every date in the data stream was used, only a subset. Instead of using the original sampling rate of 256Hz, 32Hz was used in the analysis.

Additionally, a function was created, which subsequently processes the recorded data and, depending on the parameterization and the algorithms used, can recognize certain properties in the data and output them in diagrams. These include more complex time series analyses or those that require the complete sequence as input, such as the global maxima, the baseline, or the skewness.

Communication component

The ActiveMQ-Listener from the Communication component implements the STOMP protocol and serves as the ActiveMQ client. It can communicate with the ActiveMQ broker, which holds

the topic messages for all ActiveMQ clients. Other clients include the thermal camera or the Kinect camera.

Moreover, the ActiveMQ-Listener represents an 8-state state machine that moves to the next state only with the correct control commands (see Fig. 6.1). If there are problems in one node, then the state is changed to error, and an error message is sent to the topic EMOBIKE.STATUS. Doing so ensures that the control center knows at all times in which state all nodes are located, and the controller can initiate a reset of the entire test setup, depending on the importance of a failed node.

The second class of the communication component is the CommunicationService. This class ensures that the communication channels (queues) are set up between the components and monitors their status in order to change to the error status in the event of problems.

Persistence component

The persistence component receives the individual measured values of the sensors via a data queue and stores them in a CSV file in the path known from the SessionInfo. Due to the high frequency of the incoming measured values, the data is temporarily buffered and only stored once on the data carrier at the end of a measurement.

A total of 9 different threads are started, which communicate with each other via queues. MainThread, AMQListener, CommunicationService, SensorService, DataReceiver, Graph (Plotter), Utilities (Processing), CSVWriter and DataConverter are each individual threads. These threads were necessary to enable the parallel execution of different tasks and to ensure permanent availability for control signals from the control center and data and status messages from the Plux hub (see Appendix 6.1).

2.3 Experimental Design

A laboratory test was carried out to collect the required data. A total of 25 participants, 10 males and 15 females took part in the experiment. The participants were aged between 18 and 51 years, the average age was 29 years. More details on the condition and health of the participants can be found in [MÜL2018].

The acquisition of the sensor data should be minimally inversive to the participants. So, the sensor mounting was supported by an experimenter of the same gender as the participant. This should prevent influencing the participant and not arouse any undesired emotions.

At the beginning of each trial, a training level was started to allow the participant to become familiar with the pedals, the handlebar and the brakes. Afterwards, a dynamic sequence of levels

was started as described in [MÜL+2017b] and [MÜL+2017a]. A participants emotional journey always consists of a joyful/surprising level, a fearful level and a frustrating/enraging level. Some levels do not work as intended on a participant, in such a case a second level was loaded, which should provoke the same emotion.

We decided to validate our results by a self-assessment and an observer-assessment, since the annotation of emotions is still an ongoing challenge and it is not certain whether if a self-assessment or an observer-assessment is preferable [YMG2016],

In total there are 12 levels with different purposes. To get an understanding of the visual presentation, screen-shots of the level design are provided in appendix 6.1.

The content details of the levels are described below:

Training Level

In this level the participant should get used to the ergometer and the virtual environment, therefore it is a relaxed straight ride without surprises. There were no events and therefore no emotions triggered nor evaluated.

Fork Level

If the participant is still unsafe with the handlebar, brakes and pedals, a second training level can be loaded to get used to it further. The participant can decide if he take the left road with green landscape or the right road with a desert-like hilly landscape. It is a easy ride without incidents.

Teddy Level

There is a huge teddy bear in this level and environment is enjoyable and peaceful. The participants shall rejoice as soon as they see the teddy.

Glade Level

The participant rides through a pleasant forest glade with a beautiful flora/plants and nice weather. In the middle of the level they discover a big friendly rabbit. The idea is to trigger joy in this moment.

Challenge Level

This level is demanding to the participant because he has to ride over a ramp which boosts the speed of the bicycle and comes after a bend, which makes it hard to control the jump direction.

So the most participants misses the saving platform and falling in a deep gorge. In this falling event frustration should be triggered.

Downhill Level

Here the participants drive down a narrow ramp and accelerate very strongly because the breaks are broken/disabled. At the end of the ramp there is a narrow right-angled curve where the participants fall into the lava with a high probability. It is supposed to cause frustration.

Mountain Level

In the mountain level the participant have to work harder to get up a gradient. After that a grass landscape with meter-high grass must be crossed in which large spiders lurk and jump at the driver as soon as they get too close. At this moment fear should be triggered.

Forest Level

Here the participant rides trough a gloomy forest with a bicycle lamp as the only light to see anything. In the middle of the level a jump scare event is triggered and several monsters appear directly in front of the bike. In this shocking moment fear should be triggered.

Explosion Level

This level takes place in a desert which looks like a battlefield with lots of destroyed military equipment. During the ride, explosions occur again and again combined with a loud bang/boom and if the driver is too close to an explosion, the player dies and the level starts again from the beginning.

Cliff Level

The participant is cycling along an icy mountain path at great height. There are many tight spots and curves, so you have to ride the bike very carefully and sensitively. In this level rock avalanches can be triggered to increase the difficulty and stress the participant additionally. Furthermore there is an extremely narrow bridge at the end of the level, where the most of the participants fall down the mountain. It is supposed to cause frustration, because the level starts from the beginning.

Frozen Sea Level

The participants rides on a frozen sea and collecting coins in this level. The ice is smooth and the participants slide past it if they do not drive carefully. The goal is to trigger joy for every coin reached. In contrast to this, there is another event with which malicious/wicked snowmen are triggered and appear with a loud bang in front of the participant.

Tree House Level

The level consists of a wooden plank path that has a steep gradient. If you do not accelerate strongly at the beginning and drive up with momentum, you will not make it to the top. Since nobody knows it at the beginning, it is exhausting and frustrating at the same time.

Level	**Event**	**Expected Emotion**
Teddy Level	Teddy seen	Joy
Glade Level	Rabbit seen	Joy
Mountain Level	Spider attack	Fear
Forest Level	Zombie attack	Fear
Explosion Level	Explosion	Fear
Frozen Sea Level	Snowmen	Fear
Challenge Level	Player falling	Frustration
Downhill Level	Player died	Frustration
Treehouse Level	Player stopped	Frustration
Cliff Level	Player falling	Frustration

Table 2.3: Expected emotions and levels

Over the entire period of time, data is collected from the participant and then it is determined how it changes during such an event in order to draw conclusions about the perceived emotion.

In each level the intensity of the emotion was different. So the sensor data shall differ also in expression. There was also a habituation effect to an event when it was repeated several times. For the data set for later use, several levels with different events but with the same desired emotion were combined into the expected group to get more data.

In addition to the physiological data collected for this master thesis, video data and thermal data were also recorded.

For further informations about the emotional journey and technical specifications of the EmotionBike please refer to the publications [MÜL+2015], [MÜL+2016].

2.4 Problems and Solutions

The standard pre-gelled and self-adhesive disposable electrodes for the EDA sensor at the fingers were loosened when sweating too much. So they were replaced with convenient velcro strips electrodes and applicate around the fingers. For a better hold and better connection to the skin special conductive and adhesive paste was used.

The previous positioning of ECG sensors directly at the heart on the left breast was discarded and replaced by the Eindhoven arrangement, for which new cables had to be ordered. It was less rejected by the subjects and additionally provided qualitatively better data.

The original plan of a real-time analysis of the data was abandoned, because due to the complexity and thus high CPU usage, various sporadic errors occurred, such as sudden disconnections, freezes of the visualization application or different threads in the system. Therefore, the only analysis function implemented was a peak detection mainly for EDA.

3 Related Work

3.1 Emotions

In the field of emotion recognition it is important to clarify in advance what is understood by the term *Emotion*. The following questions are therefore of central importance: How many emotions are there, what characteristics do they have and how can they be distinguished from one another?

In general, emotions are short-term but intense sensations and can be distinguished from long-term moods. In addition, emotions have a clear and discernible trigger and can have a positive or negative effect.

A more specific definition is from Scherer [SCH1993] as follows:

> ""Emotion is [...] a response to the evaluation of an external or internal stimulus event [which] is considered important for the central needs and goals of the organism."

According to Plutchik [PLU2001] there are over 90 further scientific definitions to the term *Emotion*, namely alone from the 20th century. These are a controversial topic that has always been discussed intensively. This discussion alone would go beyond the scope of the thesis, therefore is referred to the publications of Kleinginna [KK1981] and Cowie [CSB2011].

Fig. 3.1 shows how an emotion works. According to Plutchik, emotions are human feedback loops that are initially triggered by a stimulus event (e.g. threat). This is followed by the perception of the stimulus event based on personal experience or instinct (in danger). This triggers a state of feeling (anxiety) and causes physical arousal (muscle tension). The following is an impulse for action (will to run away). This impulse is then implemented with a visible behavior (running away) and finally leads to a concrete effect of the action (decreasing threat). This process is looped until the stimulus event is no longer perceptible and relaxation takes place (in safety).

3.1.1 Emotion Models

Various models have been developed in psychology for the classification of emotions. Among the models who deals with grouping of emotions, two contrary groups have emerged.

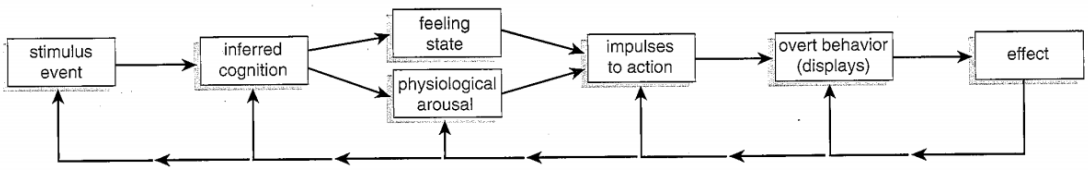


Figure 3.1: Human feedback loop (Emotion) [PLU2001]

On the one hand, these include discrete models based on a discrete set of basic emotions. One or more of these basic emotions are assigned to a person. A widespread discrete model is for example the emotion model according to Ekman [EFE1972]. It consists of 6 basic emotions (see Tab. 3.1.1). In later publications [EF1978], [SE1984] *contempt* was added as an additional category.

- Anger
- Disgust
- Fear
- Happiness
- Sadness
- Surprise
- (Contempt)

One system that uses this emotional model is the Facial Action Coding System (FACS) [EF1978], also developed by Ekman. Facial expressions are broken down into action units (AU) and categorized. In many algorithms, the "emotionless" state is also added to introduce a basic state.

Another wide spread discrete model is the *wheel of emotions* from Plutchik[PLU1962]. This model contains eight basic emotions (joy, trust, fear, surprise, sadness, aversion, resentment, expectation) between which similarity relationships exist. For illustration this can be drawn as a wheel model. From the inside out, the intensity of the emotion decreases. Similar emotions are arranged next to each other, opposite emotions opposite each other. The strongest emotions are located in the core, the basic emotions in the first ring and the weakening emotions in the following rings.

Besides the discrete models, the continuous models exist on the other hand. These do not classify emotions into categories, but map them to points in a multidimensional space. Among

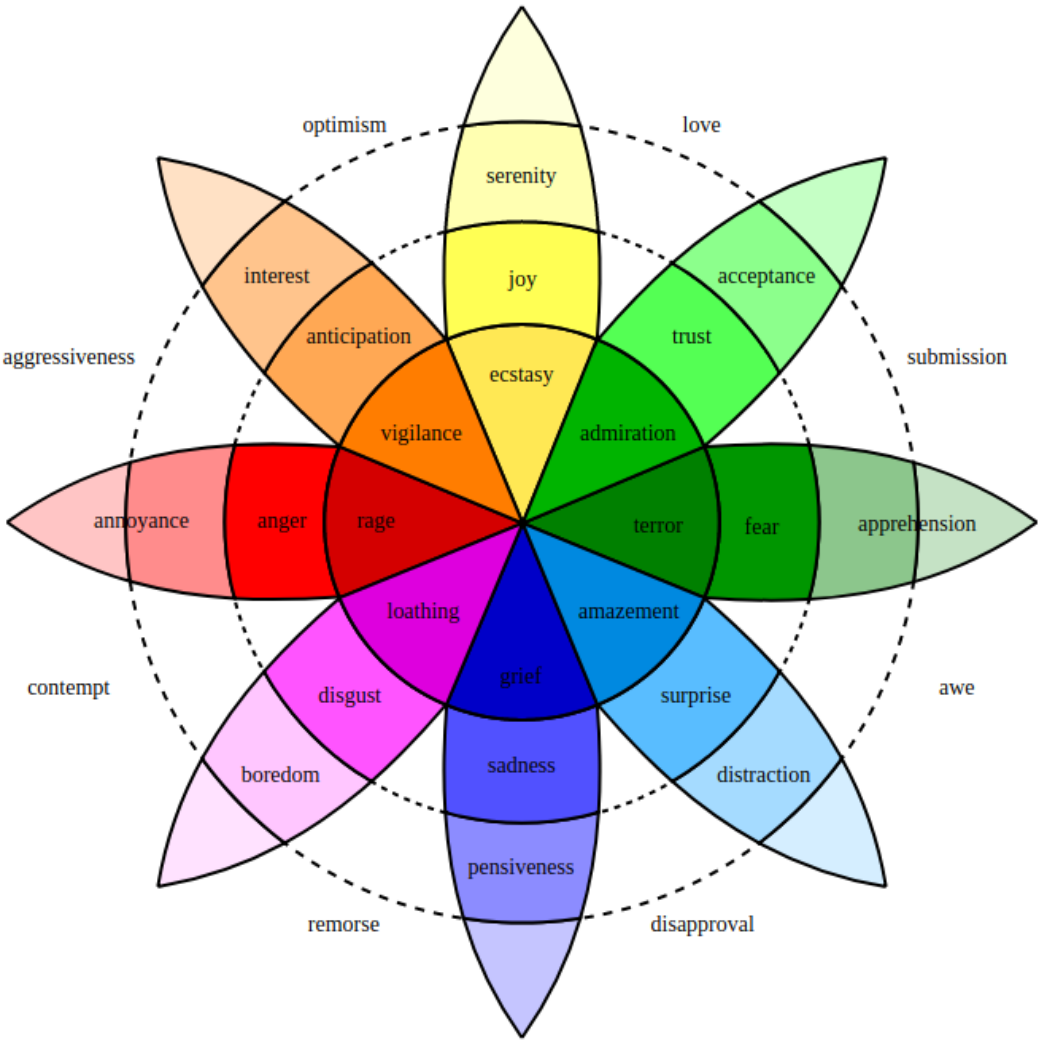


Figure 3.2: wheel of emotions [PLU1962]

representatives of this theory belongs Wundt [WUN1906]. In his view, the emotional world is not the sum of essentially unchanged elemental feelings. Rather, the wealth of qualitative emotions is inexhaustible, and thus no categorization into basic emotions is possible.

Wundt presents 3 dimensions as qualitative emotional colorations, with the following axes: *pleasure - displeasure*, *excitation - tranquilization* and *tension - relaxation*. The quality component is "pleasure - displeasure", the intensity component is "excitation - tranquilization" and the time component of an emotion is "tension - relaxation". More precisely, tension arises over time before an emotion occurs and relaxation after the emotion has occurred.

To simplify Wundt's model, according to Russel [RUS1980] the time component is omitted, and the remaining axes are excitation (Arousal) and pleasure (Valence) (cf. Fig. 3.3). This shows that emotions have been assigned to the coordinate system at certain points.

The advantage of continuous models with 2 dimensions is the simple representation and the demonstration of similarity or diversity by the spatial distance of the emotions in the coordinate system.

However, due to the continuity, it is no longer possible to clearly assign the determined coordinates to a concrete emotion, because several emotions can come into question due to the same distances.

In the EmotionBike project, Ekman's basic emotion model was originally selected, based on the above considerations. Therefore, the model is also used for the physiological data.

3.1.2 Stimulus Events

As described by Levenson 3.4 an emotion is normally activated according to the following pattern: First there is an external or internal stimulus. That means there is for example an event like a loud bang in the surrounding (external stimulus) or a thought that comes to mind (internal stimulus). The next step is a prototype-matching, the human sensory neuron is checking to see if there's been anything like this before and a certain reaction is triggered in the facial, vocal, motorical and/or autonomic system.

To classify emotions based on gathered physiological data, one needs first to be able to experimentally evoke a stimulus. Generally, there are different approaches in science to evoke emotions in a subject.

This includes, the approach using questionnaires from psychology (cf. Schwarz [SCH2007]) for example. In doing so, a participant is asked various questions one after the other, which serves as an internal stimulus and are aimed to trigger a certain emotion. The question "Where was your most beautiful holiday?", for example, triggers a positive emotion and joy in the respondent's

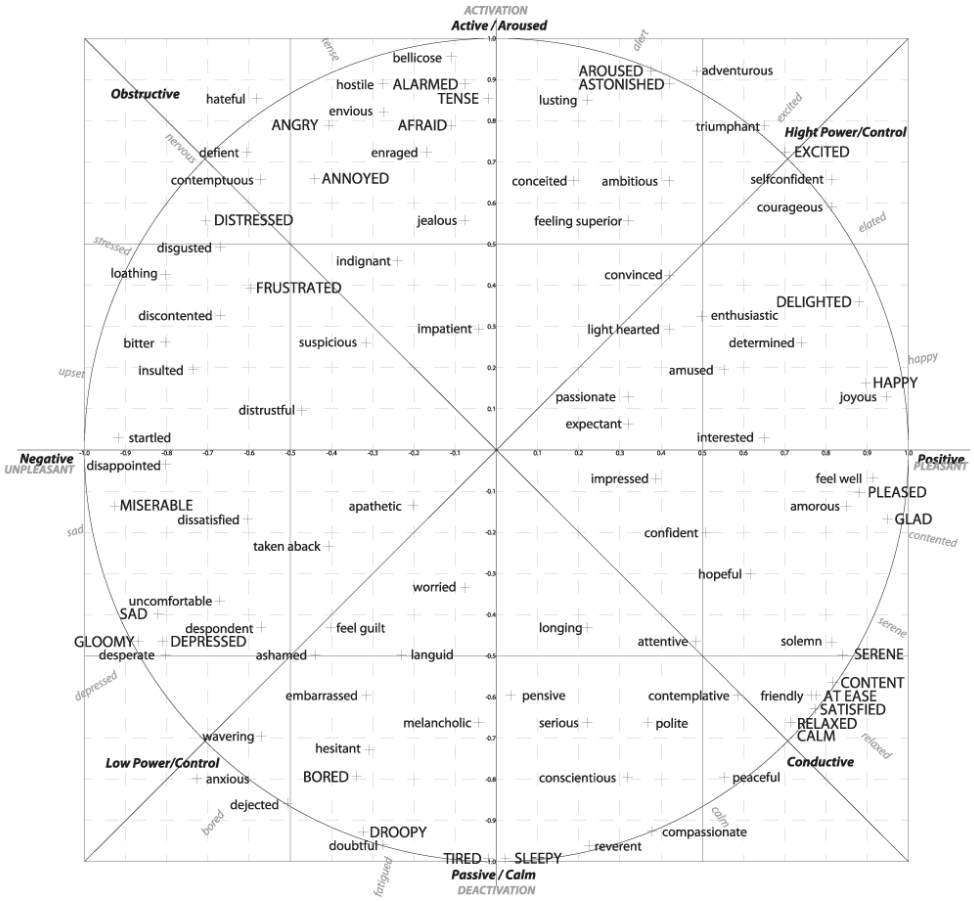


Figure 3.3: Two-dimensional emotion model according to Russel [RUS1980]

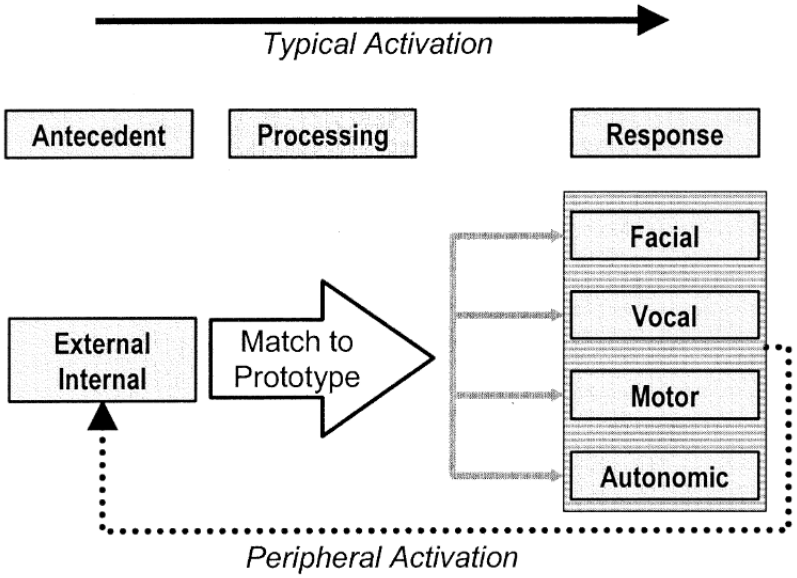


Figure 3.4: Emotion activation sequence according to Levenson [LEV2003]

memory alone. However, the interviewer should have background knowledge of the subject in order to be able to ask appropriate questions that trigger emotions.

Another frequently used approach is to play (multi-)media content such as photos, music or videos to the subject and thereby trigger emotions [SIM+1999], [JS2001].

Furthermore, a possible procedure is to let the subject act himself, either in the real world with objects and persons or in a virtual world [RIV+2007]. However, if the participants can move freely, the disadvantage is that they do not adhere to certain specifications and the measurement series are compromised as a result. For example, the physiological sensor data can be falsified by fast movements and can only be corrected with complicated calculations if the acceleration data of the movement is available.

Within the scope of the project EmotionBike the decision was made on the approach with interaction in the virtual world. The reason for this is that the participants are on the one hand tied to the provoked events, but on the other hand do not leave too much freedom of movement.

3.1.3 Emotion Recognition

Despite all the progress of intelligent systems and companion systems in recent years, emotion recognition remains the biggest problem in this area. The complexity of human behaviour makes it difficult for these systems to draw the right conclusions, and misunderstandings often arise.

A person can express his emotions in many different ways. However, according to Holodynski [HOL2009], expressions depend very much on its character, socialization and the current context, which contributes significantly to the complexity of the task.

Investigations of Mehrabian [MEH1968] revealed that emotions are spread in communication through various channels: 55% via facial expression, 38% via voice and 7% via speech (textual content).

Although facial expressions, gestures and posture play a very large role, it is difficult for video-based emotion recognition algorithms to always make reliable statements. One reason for this is that the human appearance can vary greatly and that there are different expressions in different cultures [PLU2001]. For example, the appearance can be different due to origin, social class or subculture, but it can also be distorted by injuries.

Another problem is that research into human expression was strongly focused on European and North American cultures, usually only on one social class. But also due to the increasing interest in the Asian region in the last decades the algorithms have already been improved or still have a lot of improvement potential due to the latest findings in the area of Deep Learning [LBH2015].

The second most important channel through which emotions are expressed is the voice. The means of pronunciation include pitch, speed and harmony. However, the person must also communicate his emotions orally for these algorithms to receive input data. If a person for various reasons (e.g. illness, depression, anxiety, shame) does not want to or cannot speak, then also no emotion recognition is possible with these procedures [AIG2015].

So the most promising way to detect emotions is through a persons physiological data. This is because they cannot be influenced by the person himself and are therefore very meaningful. The problem, however, is the accurate acquisition of data using bio-sensors, without noise and interference. In addition, the sensors must not restrict the person too much, as otherwise there may be influences on the emotion as such. For example, it must not happen that it is no longer the emotion of joy about an event that is in the foreground, but anger about uncomfortable sensors.

In the following, the respective "state-of-the-art" approaches for emotion recognition are presented.

Video-based approches

Video-based emotion recognition procedures use appropriate video recordings as training data, which contain one or more persons from whom certain emotions are experienced. After training, the detection rate can be determined during a test phase.

The CK+ Dataset, which is based on ideal conditions, serves as an example of such a training database. The corresponding videos show people playing emotions, once with a frontal view and once offset by 30°. This dataset documented detection rates of over 90% [CTA2015].

Since 2013 an annual competition for the best recognition rates in realistic situations, the so-called EmotiW (Emotion Recognition in the Wild) has been taking place within the framework of the ACM International Conference on Multimodal Interaction. The dataset "Acted Facial Expressions in the Wild" (AFEW) consists of video clips cut from feature films. For the 2016 Challenge the dataset was extended by reality TV clips. This should make the dataset more realistic, since it is not actors, but amateurs who embody emotions. Overall, the dataset offers a highly heterogeneous database, i.e. there are video sequences with or without camera movement, various lighting conditions, pictures in buildings or outdoors, free body movements of the person, other persons and objects in the picture, etc. It is a contrast to datasets created in laboratory environments. It should be classified into the 7 emotions according to Ekman: *rage*, *disgust*, *fear*, *joy*, *grayness*, *surprise* and *neutral*.

The best detection rates in recent years have been in the range of 50%. Convolutional Neural Networks (CNN) are mainly used for detection based on the video data. More details about the competition and the winning procedures are discussed in the corresponding publications (see [CTA2015], [WLL2015], [FAN+2016], [YAO+2016]). Further articles on video-based procedures are Bernin [BER+2017] and Corneanu [COR+2016].

Audio-based approches

Audio-based methods use the voice as the basis for emotion recognition. However, they have the disadvantage that a large part of a person's emotional expression is lost (facial expression/gesture). In addition, the way of expression is also strongly dependent on the respective language or language family. So far, two thirds of all languages have been examined for their specific characteristics. Therefore, there are no powerful methods that have a good recognition rate for all languages of the world in generalized form.

There are also various datasets for audio-based procedures, which only ever contain one specific language. In the publication of Feraru [F+2015] the 8 most common datasets are compared against each other. A feature extraction based on 31 low level descriptors (volume, harmony, etc.) and 42 statistical functions as well as a Support Vector Machine (SVM) as classifier were used. Classification was according to Arousal (positive/negative) and Valence (positive/negative). It was trained with one database and tested with another.

The result shows that the recognition rates for training and testing are approaching 100% using the same language, but drop sharply for two different languages.

Physiological-based approches

Nowadays human-machine interaction no longer takes place permanently with a terminal, but also in a dynamic environment. The rapid spread of mobile devices such as smartphones and wearables and their use in constantly changing environments has led to greater attention in research to data sources other than video and audio signals. Physiological methods have the advantage that other physical conditions (e.g. fatigue or overexertion) can also be recognized and included in the evaluation.

As Levenson [LEV2003] pointed out, emotions activates various response systems including the facial, vocal, motor and autonomic nervous system. As a result, emotions trigger subconsciously certain processes. For example, an anxiety or rage-inducing stimulus event increases blood flow to the limbs and the temperature rises as a result. In addition, respiration is influenced, blood pressure and pulse rise, sweat is secreted, which reduces skin conductance and can lead to fast movements.

For emotion recognition EEG or multi modal approaches with EEG and other peripheral sensors were often used, for example the DEAP Dataset [KOE+2011]. It tries to derive the emotions directly from the brain waves. However, EEG has the problem that it has to be positioned and calibrated very precisely and can only be used in a laboratory environment, by no means in daily life.

In the study by Nummenmaa et. al [NUM+2014] for emotion recognition using whole-body thermal images it was proven that recognition rates of 38% can be achieved. Exemplary thermal images are shown in Fig. 3.5.

In the project, the temperature sensor is attached to the left forearm, as the upper body is already recorded by a thermal camera and is therefore considered the most suitable location to obtain a further temperature value. It remains to be investigated whether the data from a temperature sensor are sufficient to obtain reliable conclusions about the emotion or whether further temperature sensors are necessary at other parts of the body (cf. Kletz and Kleimann [KK2017]).

The electrocardiogram (ECG) records the heartbeat on the basis of cardiac muscle activity. Cai et. al [CLH2009] show that in an emotion model with two emotions with the help of ECG data a recognition rate is possible in the classification of joy (to 80%) and sadness (to 90%).

Electro-dermal activity (EDA) is significantly influenced by stress and excitation and according to Haag et. al [HAA+2004] is well suited to distinguish between stressful and non-stressful situations and anger or fear. Physiologically, this is done by stimulating the formation of sweat. According to Taylor [TAY+2015] EDA values are very characteristic for emotions. Thus, the value rises very quickly and then decreases equally quickly, resulting in peaks with a time lag from the stimulus event (cf. Fig. 3.6).

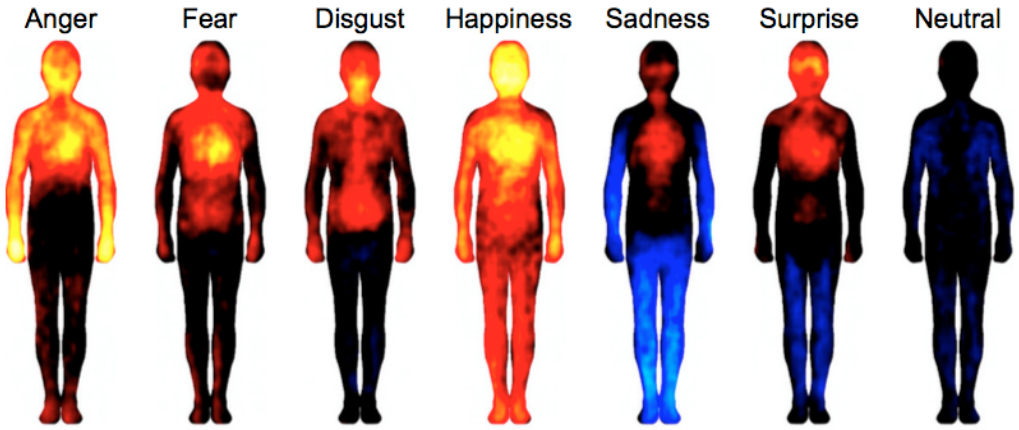


Figure 3.5: Full body thermo images for basic emotions [NUM+2014]

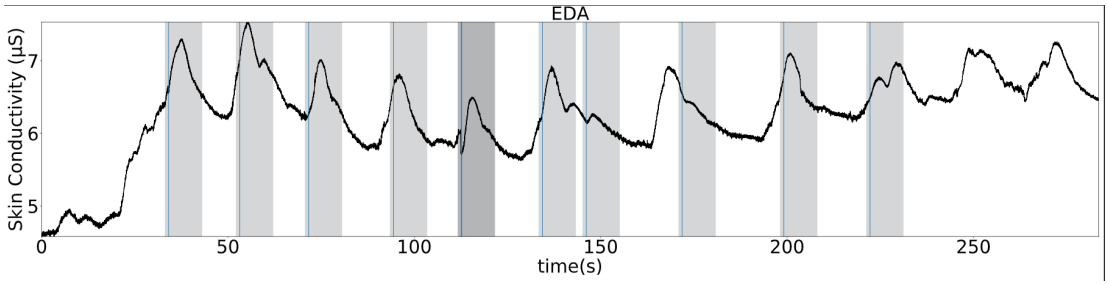


Figure 3.6: EDA-Peaks for certain stimulus events

The respiratory sensor can be used to determine how quickly or deeply one breathes. It delivers conclusions about an emotion that has occurred.

In a purely biosensor-based test setup, it would also be interesting to record muscle movements in the face using EMG sensors. However, as this would affect the video recording and the thermal camera in the project, it was not used.

For the EmotiW Challenge was planned [RIN+2015] to integrate physiological data, which was not implemented now. It should serve as another input source and improve the recognition rates of video and audio-based methods. Since it is generally difficult to generate a large dataset with physiological data, it is likely that it has been too complex to develop a suitable dataset for this challenge.

Known methods such as von Haag et. al [HAA+2004] or Kim et. al [KA2009] are equipped with a multi modal sensor network and both with a rudimentary emotion model that can classify positive and negative Valence and Arousal. Both research teams achieve detection rates of 90%. However, a further step in the process is necessary in order to depict Valence and Arousal on the individual emotions. Because in each quadrant of the coordinate system of continuous emotion models (cf. Fig. 3.3) several emotions are present and with these two methods, an exact allocation is not possible.

Ultimately, all procedures are strongly dependent on the respective training set used and the emotional model. This makes it difficult to make a direct comparison between audio and video-based methods, but also between the methods within a category. Moreover, many datasets are not representative enough or represent only a small part of reality.

Research is therefore not complete and there is still a need for meaningful datasets that contain as many data channels (audio/video or physiological data) as possible simultaneously. Therefore, great importance is attached to creating an own dataset as part of the EmotionBike project.

3.2 Time Series Analysis

In the following, the current state of research in the field of time series analysis is presented. The aim is to present several current solution approaches that are capable of processing time series data and recognizing the corresponding emotions.

To analyze the sensor data, a suitable mathematical model must first be found that reflects the most important aspects of the problem described in the previous chapter. Typically, the data supplied by a sensor is modeled as a time series. The sensor's measurement signal is recorded at regular intervals within a finite time interval, resulting in a sequence of readings after l observations:

$$x_1 \rightarrow x_2 \rightarrow \cdots \rightarrow x_l, \text{ or } (x_1, \ldots, x_l)$$

In the case of m sensors, m have different time series that must be viewed independently of each other:

$$S_{i,j} = (x_{i,j}(t_1), x_{i,j}(t_2), \ldots, x_{i,j}(t_{l(j)})),$$

where $x_{i,j}$ is the measured value of sensor j ($j \in 1 \ldots m$) at time t for the subject i with $i \in 1 \ldots n$. The length of each time series can be different for sensors ($l = l(j)$). This results in a total of $n \cdot m \cdot \sum_{j=1}^{m} l(j)$ measured values. In addition, a target vector is specified:

$$Y = (y_1, \ldots, y_n),$$

where y_i describes the condition of the subject i, i.e. his emotion.

The goal of sensor data analysis is to recognize certain *Features* in the course of time series and to assign these patterns to the emotions used. Thus in the future a statement is to be made possible for arbitrary data of a sensor, which emotion it concerns with these data.

The solution methods for the problem formulated above are assigned to the research area of **time series classification** (TSC). These procedures can be divided into the following categories:

- *shape based classification*
- *feature based classification*
- *neural network based classification*

Each of these categories attempts to classify the time series data in its own way. In the following, the categories are described in detail and the differences are shown.

3.2.1 Shape-based

Shape-based approches deal only with the shape of the time series curve and enable pattern recognition by calculating the distance between two curves in a certain way. Certain time series are predefined, whose membership to the given classes is known. Then a k-Nearest-Neighbor algorithm is used to determine to which known time series a new time series is closest and thus determines its class affiliation. The best known and most successful of the shape-based methods is ”Dynamic Time Warping” (DTW) [MÜL2007].

Dynamic Time Warping (DTW)

With the DTW, two time-dependent value-standardized signal sequences are compared, see Fig. 3.7:

$$X := (x_1, x_2, ..., x_n) \text{ and } Y := (y_1, y_2, ..., y_m)$$

Each x_i is mapped to y_j, whereby several x can be assigned to one y, or several y to one x. The method originally comes from automatic speech recognition and is intended to recognise variations through words spoken at different speeds, but it can also be used for other time-dependent data.

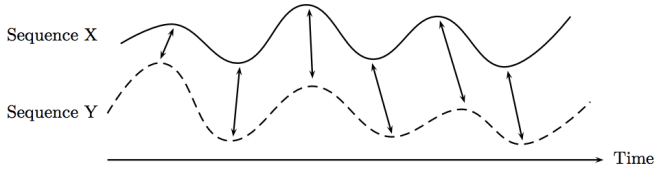


Figure 3.7: Time series X and Y [MÜL2007]

Using a cost function $c(x_i, y_j)$, points that are close to each other are provided with low costs, points that are far apart are provided with high costs. A cost matrix M is then created for the cross product of all elements of X and Y. A graphical representation of the cost matrix is shown in Fig. 3.8. Light spots in the diagram are high costs and dark spots - low costs. The aim of the algorithm is to use this cost matrix to find the optimal, most favorable path (so-called "Warping Path"). The path search starts at $c(x_1, y_1)$ and ends at $c(x_n, y_m)$. Restrictions for path search are a monotonous increase and a fixed maximum increment of 1.

The resulting path is thus a list of individual costs, which adds up the total costs. With the help of these total costs it can now be determined how similar two signal sequences are. In principle, the algorithm thus performs adaptive time normalization. This means that the signal sequence is stretched or compressed.

DTW provides the best results for time series classification on many benchmark datasets according to [CKF2016]. However, it is very memory intensive and scales poorly with increasing time series length [BAG+2017].

3.2.2 Feature-based

Feature-based methods extract a large number of generic features from each time series, which are used to classify the time series. Thus it is distinguished under [NAM2001]:

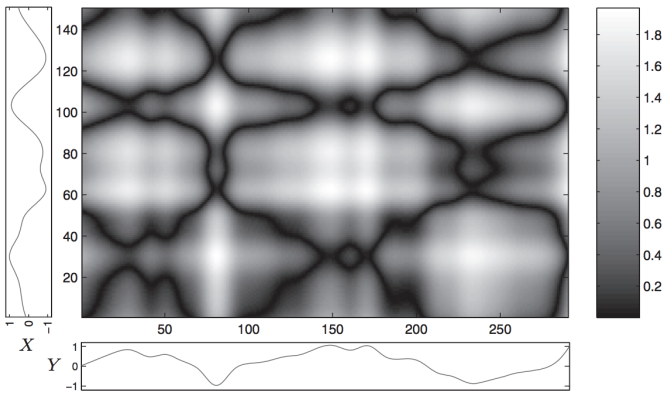


Figure 3.8: Cost matrix for $X \times Y$ [MÜL2007]

- First-order features - calculated directly from the original time series.
- Second order features - original time series must first be transformed, e.g. as follows: $f(t) = f(t + D) - f(t)$. where f is the value function of the time series, t the time and D a user-defined value. The results are features that are based on the difference in time and thus map local dependencies.

According to [CKF2016] there are about 1000 different possible features for the time series analysis, of which mostly about 50 are used. Examples include the artifact mean, standard deviation, skewness, curvature, median, number of peaks with a certain increase, periodicity of peaks, global trend, etc.

Feature extraction serves to reduce dimensions, since each time series can be represented by a set of statistically significant features. The feature vectors can then be classified using classification algorithms such as AdaBoost or Random Forest. This means that it is no longer necessary to consider every single point in the time series for classification, but only the calculated features.

AdaBoost is a boosting method published by Freund and Schapire [FS1995] that combines many weak classifiers into one strong classifier. The Random Forest method [BRE2001] is based on many independent and randomly generated decision trees. The decisions from all decision trees are collected during classification and the most frequently selected class is returned as the final result.

The advantage of feature-based classification procedures is that they are significantly faster than shape-based procedures, since only the calculated features have to be considered for classification, rather than each individual data point in the time series. The big disadvantage, however, is

that it takes a lot of effort to select the right features for a particular problem and possible data pre-processing is necessary.

3.2.3 Deep Learning-based

Direct methods such as neural networks or other deep learning models automatically learn the features from the time series. Due to the constantly increasing amount of computing capacity, neural network-based classification methods are becoming more and more powerful and are now among the preferred methods, especially for very large data volumes. Typically network classes such as ”Long Short-Term Memory” (LSTM) or ”Convolutional Neural Networks” (CNN) are used for time series.

LSTM networks were first described in 1997 by Hochreiter and Schmidhuber [HS1997] and are based on Recurrent Neural Networks, which are especially able to recognize temporal relationships in data series.

The classic area of application for CNNs is object classification by feature extraction using 2D or 3D images. In this area they have become the standard and provide state-of-the-art object detection rates. However, since time series are one-dimensional as opposed to images, the CNNs would have to be adapted to 1D data. There are already first successful attempts, see [ZHA+2017] and [WYO2017].

The big advantage of the Deep Learning models is that the features are no longer defined generically or manually, but are learned by the network for each specific problem. This means that these models can describe the problem more precisely than the other two categories of time series classification procedures. However, they usually require a great deal of training data, which is often a great challenge in practice. If too few training data are available, many adjustments or optimizations are necessary to obtain good results.

4 Proposed Algorithms

It is hypothesized that with physiological sensor data alone an emotion recognition according to the model with 7 basic emotions by Ekman is possible. For this purpose, a test setup will be implemented which will provide the required data. 20 subjects will produce the data in order to obtain a sufficiently large dataset. The data obtained is interference suppressed, filtered and normalized. It will then be used for training and testing an even more specific process.

In addition, the overall EmotionBike project aims to increase the possible accuracy of emotion recognition by using physiological data as an additional data source for decision making. If the video data do not allow a clear assignment, the physiological data can be the decisive factor in the classification.

After completion of the experiment, the data should be evaluated by comparing the emotions determined by the respective classifier (video, physiological data) with the expected emotions.

The approach is to use an ensemble in order to eliminate the uncertainty of choosing which channel/sensor is the most reliable source to predict emotions. Furthermore, ensemble learning allows us to combine all the available sensor information under the assumption that the latter produces higher recognition rates than any of the sensors alone. [KAM+2018]

We propose two machine learning and five deep learning models trained in ensemble to classify the physiological time series data. The machine learning algorithms are DTW with k Nearest Neighbor Classification (shape-based method) and Feature Extraction with SVM Classification (feature-based method). The proposed algorithms from the deep learning domain are well-established models, which have already been used for time series classification with many benchmark datasets as described in [WYO2017]. The first model is a multilayer perceptron (**MLP**), the second is fully convolutional network (**FCN**), the third is residual network (**ResNet**), the fourth is Long-Short Term Memory (**LSTM**) and the fifth is (**Autoencoder**).

In the following we will explain the approaches in detail.

4.1 DTW with kNN

As mentioned in Chapter 3.2.1, DTW belongs to the shape-based methods for time-series classification. As DTW is one of the most successful methods, which gives a strong baseline performance, we have decided to implement it and compare with the other algorithms.

Fig. 4.1 illustrates two different univariate time-series. On the right the resulting warping path is shown. Evidently, the distance between the two time-series is relatively small, so that the kNN will classify one of these as belonging to the same class as the other.

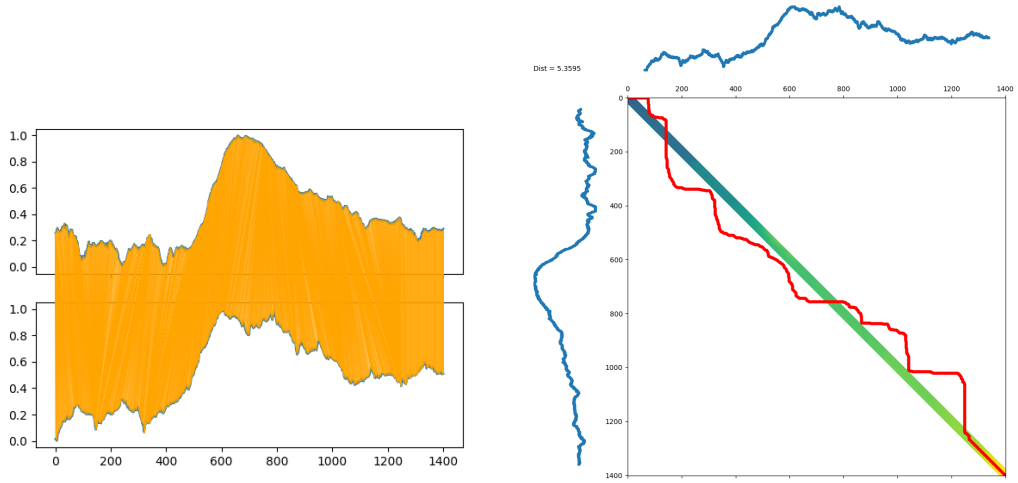


Figure 4.1: Left: Warping of a time series - Right: Warping path with distance

4.2 Support Vector Machines

A big advantage of SVM is the use of different kernels to increase the generalization performance and therefore the classification speed. There are nowadays a variety of different kernels developed. We have used the two most common kernels: linear and radial basic function (rbf). A linear kernel has an advantage if the data set is linearly separable. Linear separability means, that two groups of data points in a n-dimensional space can be separated by an (n-1)-dimensional hyperplane.

There are two hyperparameters for the SVM. The error weight (C) and the kernel parameter γ. The C parameter controls the cost of misclassification on the training data. Where a small value for C means higher bias and lower variance, high values means lower bias and higher variance. In turn, γ is a kernel parameter that determines how much and how far the influence of a support

vector is for the classification of a single training example. Where a small γ means the support vector have far influence and a high values means close/nearby influence.

4.3 Multi Layer Perceptron

The most common type of deep learning models is the Multilayer Perceptron (**MLP**). It belongs to the class of feedforward artificial neural networks and is composed of several layers of neurons. The network architecture consists of one input layer, one output layer and at least one intermediate layer (also known as hidden layer). The term "feedforward" means that the information flows through the network without cycles. An MLP is also fully connected, as seen in Fig. 4.2, i.e. every node is connected to all the nodes of the next layer. Adding hidden layers enables the network to learn higher-order features out of the input data [HAY+2009]. Traditionally, MLPs are trained in a supervised manner with backpropagation algorithm, which is a special variant of gradient descent [LEC+1998].

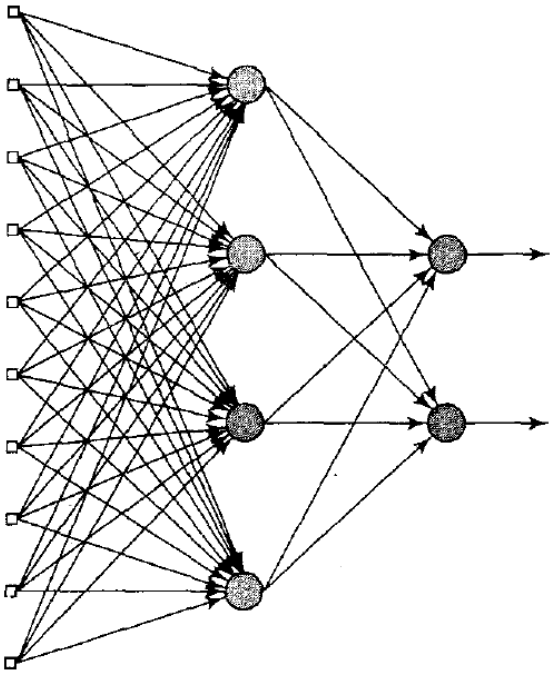


Figure 4.2: MLP with one hidden layer [HAY+2009]

One important characteristics of MLPs is that the each neuron uses a non-linear activation function in order to calculate its output value. The presence of these non-linearities distinguishes

MLP from a single-layer perceptron [HAY+2009]. In the literature, there is a variety of commonly used activation functions, like sigmoid function f_1 or hyperbolic tangent f_2:

$$f_1(x) = \frac{1}{1+e^{-x}}$$
$$f_2(x) = \tanh(x)$$

However, scientific evidence suggests that these so-called "saturating" non-linearities are prone to suffer from vanishing gradient problem (see [KRI2014] and [CUH2015]). The consequence is that the training of larger networks slows down dramatically, leading to the necessity of using other non-linearities such as **ReLU** ("rectified linear unit" [KRI2014]):

$$f_3(x) = max(0, x)$$

In order to fulfill any classification task, an MLP learns all the connection weights during the training phase. The goal of training is to find the weights which produce the lowest classification error possible on the given dataset. The so-called *loss function* measures the error and is used as a function being optimized during the training [LEC+1998]. We used **categorical cross-entropy** as the loss function (also known as logarithmic loss). It gives a probability value between zero and one. It is a performance measure where zero loss would be the best possible model. In such case, the last network layer is commonly a **softmax layer** where the number of neurons equals the number of classes in which the dataset should be separated.

As mentioned above, the training/optimization algorithm is back-propagation, which consists of two distinct passes of computation [HAY+2009]. During the forward pass, the current weights are used in order to calculate the network outputs layer-by-layer for the given input data sample. Then, the error of classification can be determined, as each training sample has a known class label ("ground truth"), which can now be compared to the network output. In turn, the backward pass propagates the error signal starting from the last layer until the first, at each step calculating the gradients. Thus, the change of weights ϕ can be performed recursively using the following formulae [SUT+2013]:

$$v_{t+1} = \mu \cdot v_t - \epsilon \cdot \nabla f(\phi_t)$$
$$\phi_{t+1} = \phi_t + v_{t+1}$$

where $\epsilon > 0$ is the learning rate, $\mu \in [0, 1]$ - the momentum coefficient and $\nabla f(\phi_t)$ is the gradient of the objective function f at iteration t. These equations differ from classical formulae of gradient descent [LEC+1998] by the momentum coefficient μ, which is responsible

for smoothing the gradients that are otherwise very sensitive to signal noise. According to the equations above, the gradient descent is accelerated by accumulating the velocity vector v_t in the directions, where the gradient is constantly reducing [SUT+2013].

In our work, we employed the **Adam** algorithm [KB2014] which was shown to outperform the commonly used optimizers for network architectures such as MLP, CNN and Autoencoder. It calculates the adaptive learning rates out of gradient moments, at the same time demanding little memory resources. Adam combines the advantages of other well-known optimizers like RMSProp and AdaGrad, while covering their disadvantages (poor results with sparse gradients, divergence due to large initialization bias).

One of the major problems with MLPs is their tendency of *overfitting*. This means that if the network architecture is too complex for the given dataset, the model adapts too much to the dataset (eg. after too many training iterations) and is not able to classify any data that differs from the original samples. In such cases the model is said to generalize the data too poorly [HAY+2009].

There are several ways to avoid overfitting and to produce a model with better generalization ability. One widely-used technique of regularization is called Dropout [SRI+2014] where a particular percentage (the dropout rate) of neurons are deactivated (i.e. their outputs are set to zero) at each training iteration. The principle of Dropout is illustrated in 4.3. This technique forces the network to learn more robust features from the data, as each neuron cannot fully rely on its previous counterparts [KRI2014].

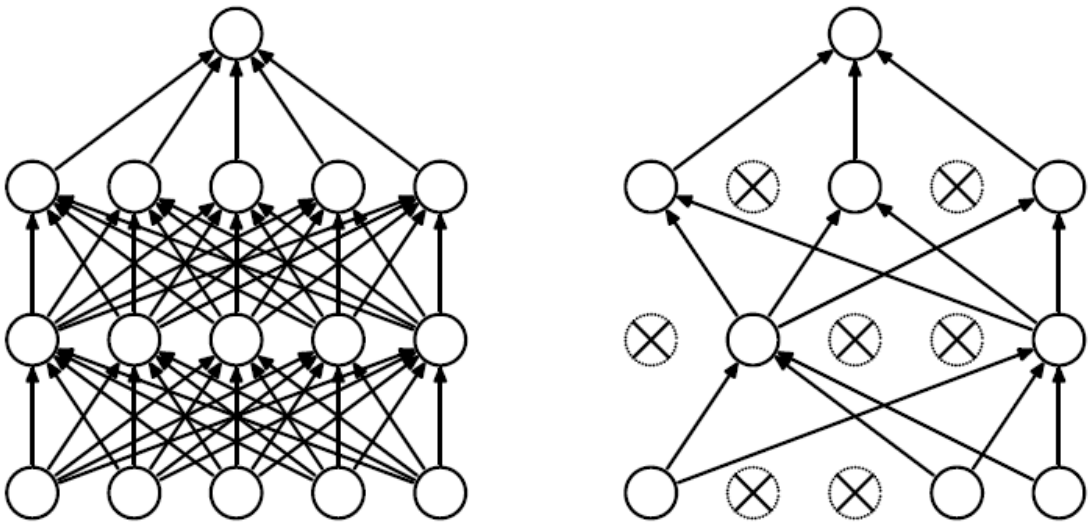


Figure 4.3: Left: MLP without Dropout - Right: MLP with Dropout [SRI+2014]

Another approach to avoid overfitting is to regularize the MLP by means of weight decay stepping [NK2009]. Regularization is a technique which helps to adjust the complexity of the data with that of the model. This technique is utilized in order to handle high correlation between the features, filter the noise out of the data and by doing so it avoids overfitting. The idea is to introduce an additional penalty term in the objective function which penalizes the extreme weights of the network. The most common types of regularization are the so called L1 and L2 regularization which calculate the L1 and L2 norm of the weights, respectively, and use it as a penalty term.

4.4 Fully Convolutional Network

A Fully Convolutional Network (FCN) is a special type of Convolutional Neural Network (CNN), which in turn is an advanced MLP. CNN is one of the most extensively used models in the deep learning domain, as it achieves state of the art performance, in some areas already outperforming humans (eg. [IS2015]). Nowadays, CNNs are able to solve the wide variety of problems like image and video classification, object detection and segmentation, text classification, speech recognition etc.

One of the distinctive characteristics of CNNs is their inherent invariance to scaling, rotation and translation of input data [HAY+2009]. This invariance is achieved by utilizing specific types of neuron layers like convolution layer, pooling layer and fully-connected layer (cf. [MAH+2017]). An example of the representative CNN network architecture is shown in Fig. 4.4. In this case, the input data is a 2D image with a certain object in it, and the output is a vector of probabilities of this object belonging to each of the given object classes/categories.

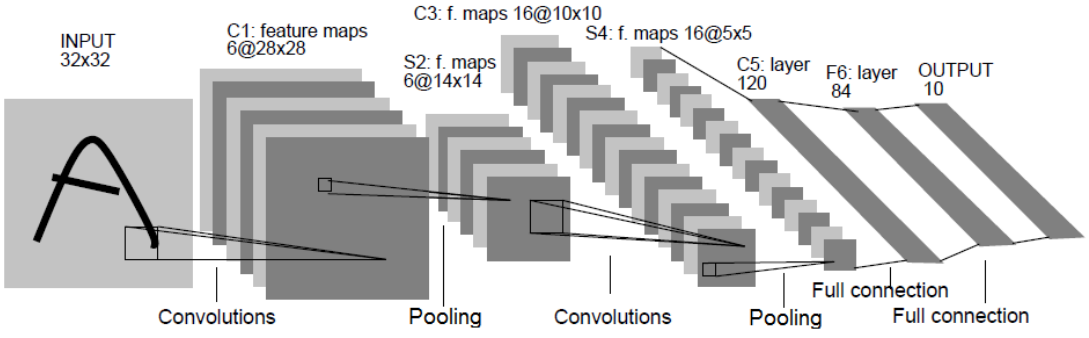


Figure 4.4: CNN network architecture [LEC+1998]

The neurons of a **convolution layer** are organized in planes, where the neurons of the same plane share weights. Each plane is responsible for extracting one certain feature from the

incoming image, whereby the output of each plane is called "*feature map*" [LEC+1998]. Weight sharing substantially reduces the overall amount of trainable parameters (weights) in the network as compared to an MLP. Each unit of one plane receives its input from a certain small local area of the previous layer. The size of this area corresponds to the size of the unit's weight matrix, is fixed in advance and called "*filter size*". Thus, a feature can be detected independent of its precise position, whereas the feature position relative to other features remains approximately the same [HAY+2009]. Additionally, all neurons of the convolution layer have the same activation function, eg. Sigmoid or ReLU.

Pooling layer follows the convolution layer and is designed to reduce the size of the feature maps by means of local averaging, so that the output is less sensitive to object distortions [LEC+1998]. Here, the neurons are also organized in planes, but each unit performs averaging of the values within one local area of the previous feature map. Accordingly, the pooling layer also has to define the filter size, which is typically very small.

By stacking the combination of convolutional and pooling layers on top of each other (as in Fig. 4.4), a CNN is able to extract feature hierarchy from any given data, meaning that the complex features extracted in the last layers (in case of image data – rectangles, circles) are built out of the basic features extracted in the previous layers (lines, edges, arcs). While the size of the feature maps decreases, their count grows in order to capture the possible feature combinations [HAY+2009].

After these feature extraction layers follows the co-called "*classifier*", usually an MLP with few fully connected layers, which is responsible for classification based on the resulting features. Because the feature map sizes change after each layer, the transition between the feature extraction layers and the classifier should be carefully designed for one specific input data size. Thus, a typical CNN works with fixed data size, which can often be a restriction for some real world datasets.

The idea behind **FCN** is to construct a neural network that does not include any pooling or full-connected layers. This allows end-to-end training from arbitrary-sized inputs [SLD2016], covering the one shortcoming of CNNs mentioned above. Another benefit of FCNs is that they keep spatial coordinates of the features, see Fig. 4.5. This makes them a good choice for the problems where this information can be highly relevant.

Indeed, FCN was originally introduced for semantic segmentation of images [SLD2016], but have now also been used successfully for time series analysis [WYO2017]. For TSC, the main advantage of FCNs is that the time series length is kept unchanged throughout the convolution layers. Most notably, one dimensional convolutional layers are used in FCNs for this purpose, since the univariate time series are 1D vectors (as opposed to 2D images).

On the one hand, not using a classifier in an FCN means getting rid of the layers with the most trainable weights that are also mainly responsible for overfitting [SRI+2014]. On the other hand, calculating class membership out of a set of the final feature maps becomes a challenge. One possible way to achieve this is to apply **global average pooling** [LCY2013]. This technique generates a 1D vector by calculating the average value per final feature map. Not having any additional parameters, global pooling forces feature maps to become class confidence maps. The output vector can be directly fed to the softmax layer [WYO2017].

Another useful technique to handle FCNs is called **batch normalization** (BN) [IS2015]. Its effect is similar to dropout, the latter not being used in FCNs due to the lack of full-connected layers. Batch normalization layer obtains its inputs directly from the output of a convolutional layer and normalizes them over the batch of different input samples, before the non-linearity like ReLU is applied. Thus, data normalization which is normally executed only once is now part of the network architecture. BN is rather widespread method in context of batch training algorithms (eg. Adam, see previous chapter), because it was shown to cover up for the poor choice of learning rate and bad weight initialization. It also outperformed other regularization techniques in several classification tasks, while requiring considerably less training iterations [IS2015].

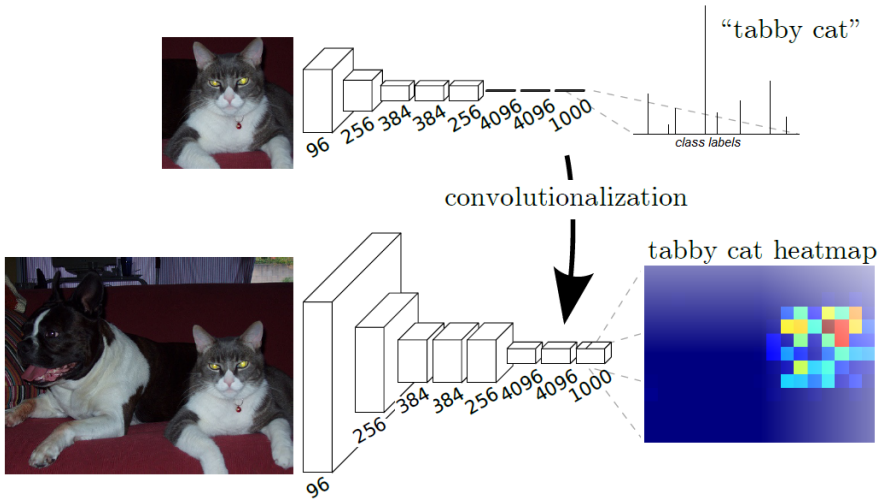


Figure 4.5: Difference between a CNN (above) and an FCN (below) [SLD2016]

4.5 Residual Network

Residual Network (**ResNet**) was originally proposed by [HE+2016] to overcome the problems of training very deep neural networks with hundreds of layers. For this purpose, shortcut connections

are introduced that perform identity mapping without added complexity, resulting in deeper models and better accuracy. An example of such shortcut connection is illustrated in Fig. 4.6.

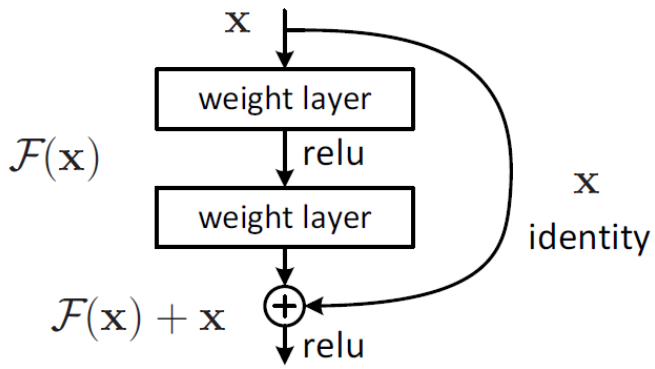


Figure 4.6: Building block of a ResNet [HE+2016]

Accordingly, a ResNet contains a sequence of building blocks, each of which includes few convolutional layers (typically 2-3). The input signal of the block is simply added to the output of the last convolutional layer, without adding more complexity to the network. In fact, the authors of [HE+2016] were able to construct a network whose complexity was even lower than that of a competitive CNN, at the same time having 152 layers – 8 times as many as in the said CNN.

Analogous to FCNs, a common practice for ResNets is to avoid dropout, make use of batch normalization between each convolutional layer and ReLU, as well as to use global average pooling layer instead of the classifier (cf. [HE+2016], [WYO2017]).

4.6 Long Short Term Memory Network

The Long Short Term Memory Networks (**LSTM**) was originally developed by Hochreiter and Schmidhuber [HS1997] as a special type of Recurrent Neural Networks (**RNN**).

RNNs differ from feed-forward neural networks in that they introduce time into the model [LBE2015]. Thus, RNNs can naturally handle sequential input data. It is achieved through recurrent connections, i.e. connections between nodes representing the adjacent time steps or even self-connections across time. Fig. 4.7 shows a simple example of an RNN, where the signal passes from bottom to the top at each time step, whereas the recurrent connection between the time steps is the dashed line.

Such RNN may be interpreted as a feed-forward neural network which can be unfolded in time, as depicted in Fig. 4.8. In this case, the network is broader, but evidently shares weights

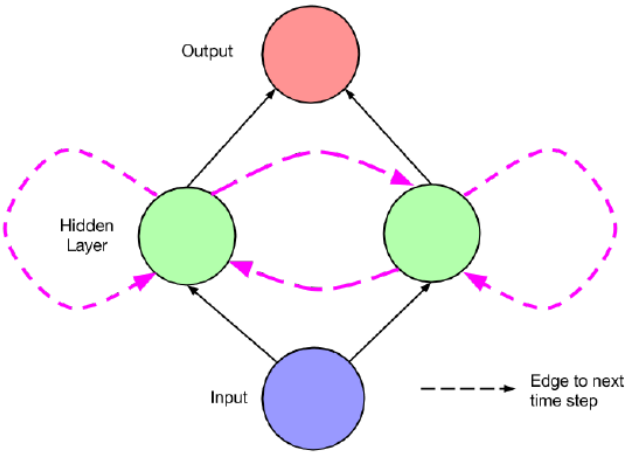


Figure 4.7: Simple RNN [LBE2015]

across time steps. RNNs are mostly trained using a special case of Backpropagation algorithm, namely "Backpropagation Through Time" [LBE2015].

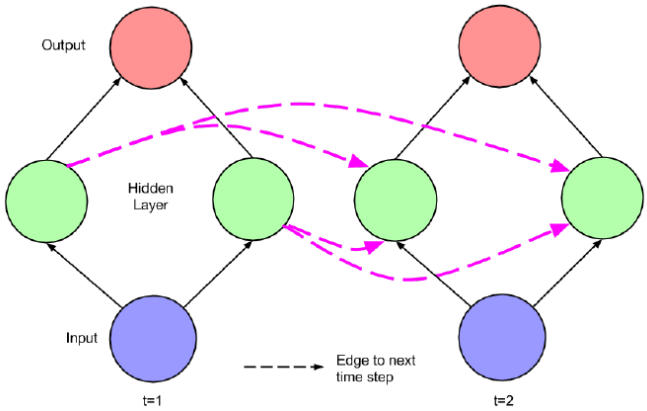


Figure 4.8: RNN unfolded in time [LBE2015]

LSTM is an improvement of RNNs, which are known to be very susceptible to the vanishing gradients problem [KAR+2018]. It means that the error gradient can be infinitesimal if the values are being multiplied many times by small numbers, which occurs when the input changes only slightly across many training iterations. LSTMs can prevent this problem by means of so-called "memory cells" which remember values for longer time steps.

A memory cell is actually composed from several basic nodes (see Fig. 4.9), some of which perform multiplication denoted by Π. The input node of the memory cell at the time step t,

$g_c^{(t)}$, is a weighted sum of the input data and the output of the hidden layer at the previous time step $t-1$. The input gate $i_c^{(t)}$ acts the same as the input node, but additionally applies sigmoid function σ to the resulting value. The internal state $s_c^{(t)}$ has a recurrent self-connection with constant weight, which prevents gradients from vanishing. It applies linear activation function, denoted by backslashed line. The output gate $o_c^{(t)}$ is then multiplied with the internal state to produce the final output value of the memory cell $v_c^{(t)}$. For further description and formulae cf. [LBE2015].

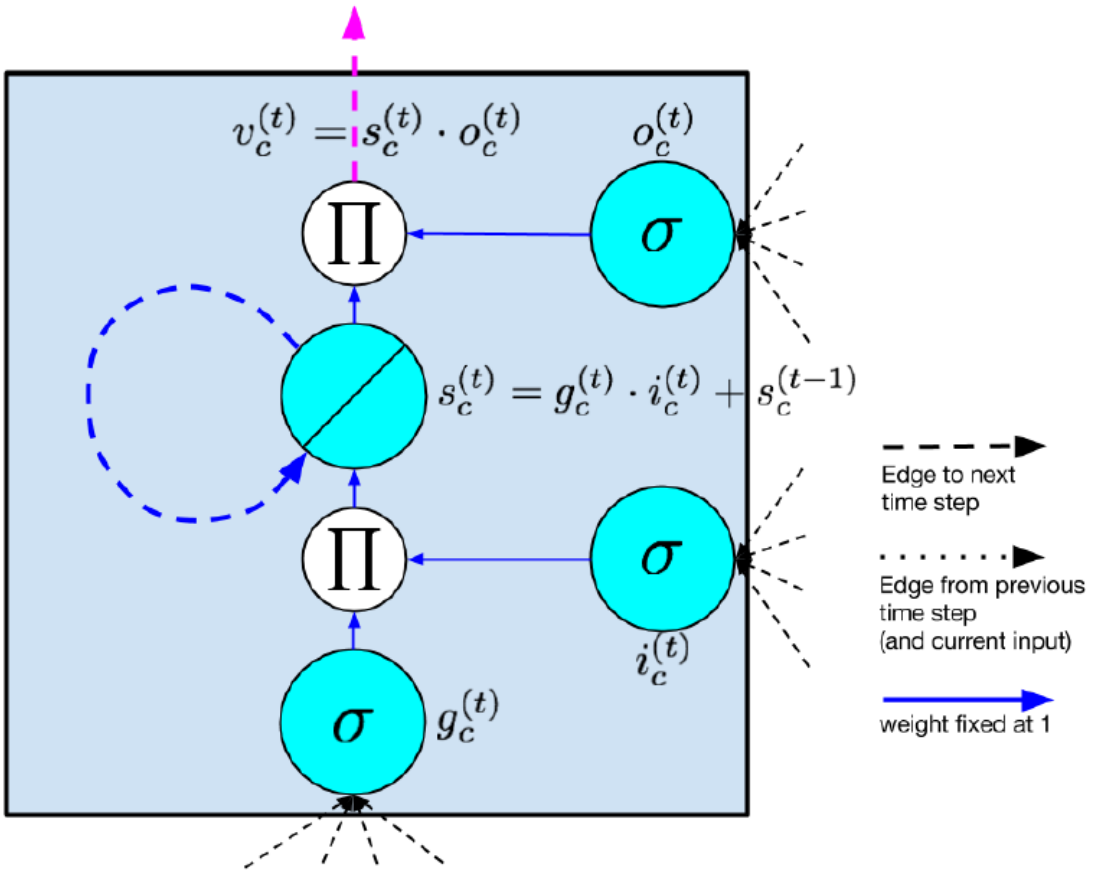


Figure 4.9: LSTM memory cell [LBE2015]

Nowadays, LSTMs are successfully applied to many tasks where the sequence of input data is essential, such as machine translation, image and video captioning, dialogue systems, handwriting recognition, time series analysis [KAR+2018], human activity recognition [WAN+2017] etc.

4.7 Autoencoder

The last of the proposed models is **Autoencoder**, a kind of artificial neural network. Autoencoder is specifically designed to learn a latent representation of the input data within its hidden layers. The learning takes place in two steps, see Fig. 4.10 . The encoder network transforms the data, typically into a vector with lower dimensions. Subsequently, the decoder network attempts to reconstruct the original data out of this encoding. These networks are combined in order to be trained with back-propagation, where the loss function is the reconstruction error [ZAH+2018].

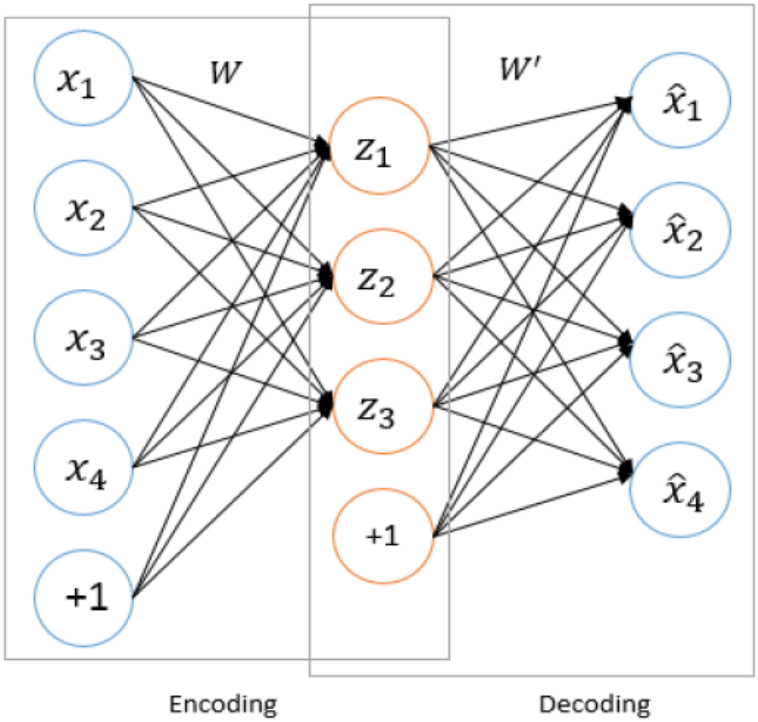


Figure 4.10: Autoencoder which learns latent vector (middle nodes) [ZAH+2018]

Generally, both encoder and decoder network consist of more than one hidden layer, they can even have complex structure like CNN, FCN, RNN etc. Autoencoders are mostly advantageous for dimension reduction, feature extraction as well as representation learning in an unsupervised manner [WAN+2017].

4.8 Network Ensembles

All the deep learning models described above were shown to achieve state of the art performance in many univariate TSC datasets [WAN+2017]. However, our problem setting with physiological signals is defined as multivariate TSC (see Chapter 3.2). Therefore, in order to apply these models to our case, we need to find a way to utilize each source of data (sensor) simultaneously. This challenge is known in research as *sensor fusion*.

Our problem setting is complicated by the fact that the given data is spread across different channels, each of which is produced by one bio-sensor. It implies using some kind of data fusion. The authors of [GAH2017] faced the similar problem with different data channels (namely video, audio and text) for the deception detection task. Their solution was to test two general fusion mechanisms. The first mechanism, *feature-level fusion*, concatenates the extracted features before feeding them to a classifier. The second, *decision-level fusion*, employs several so-called "weak learners" that learn features from each data channel separately. Afterwards, the class predictions are calculated out of all weak learners' outputs, which corresponds to the general ensemble learning procedure. Similar comparison of data fusion mechanisms was done in [ZHO+2017], where the goal was to classify physiological time series data for emotion recognition. The results have shown the superiority of the decision-level fusion, which we have therefore chosen for our experiments. [KAM+2018]

As stated above, the proposed deep learning models were used to classify one data channel at a time. In order to combine the class predictions coming from each data channel, an ensemble approach is now introduced.

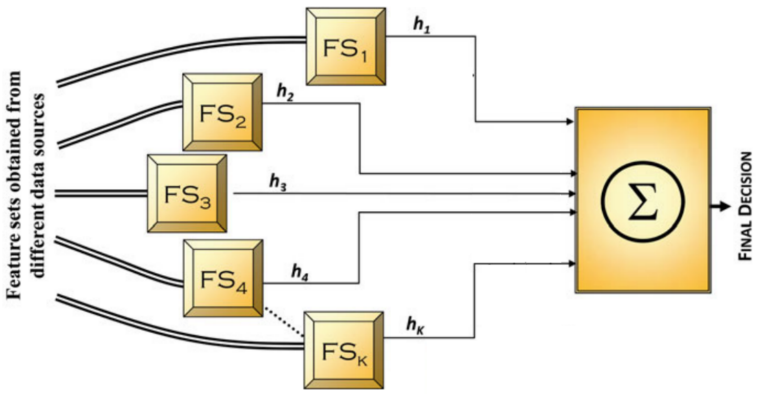


Figure 4.11: Model ensemble [KAM+2018]

Each ensemble in the following experiments contains 5 equal models working with its own data channel. The first ensemble consists of MLPs, the second – of FCNs, the third – of ResNets, the fourth – of LSTMs, and the fifth – of Autoencoders. The entire ensemble architecture is illustrated in Figure 4.11. The single models are represented with FS_i blocks, their outputs h_i being averaged to produce an ensemble output.

5 Experiments

5.1 Setup

In order to achieve the goal to classify physiological data with respect to human emotions a computer setup was configured. All implementations are made in Python programming language. As framework to implement the proposed deep learning models keras [1] and as a backend the machine learning platform tensorflow [2] was used. Additionally different functions for classification, preprocessing and model selection are taken from scikit-learn [3]

The training and tests was run on a workstation with Intel Core i7-6700K (4 core / 8 threads), 16 GB DDR4 RAM, a NVIDIA GeForce GTX 1080 GPU with 8 GB VRAM on the operating system Ubuntu 16.4.

5.2 Dataset

All time series of all events, which should cause fear or joy, serve as the data set(see Table. 2.3). This was done after the trials with the participants by selecting the representative time series and the corresponding emotion labels.

In Table. 5.1 you can see a typical human reaction to a scaring situation. At the end of the mountain level, there are large spiders in the gras and attacks the participant (each blue line is a spider attack). Were the first two spiders still caused irritation and disbelief, an escape behavior occurs from the third spider onwards. You can see that the EDA value is rising, so sweat is secreted more often. The temperature is falling and the BVP signal amplitude lowers. This can be explained by the fact that the blood supply to the extremities is reduced by the body and the hands become cold.

Due to the noticeable variations in psycho-physiological qualities of the participants, we normalized each time series channel-wise, subtracting the minimal value and dividing by the value range, as in [ZHO+2017].

[1] https://keras.io/
[2] https://www.tensorflow.org/
[3] https://scikit-learn.org

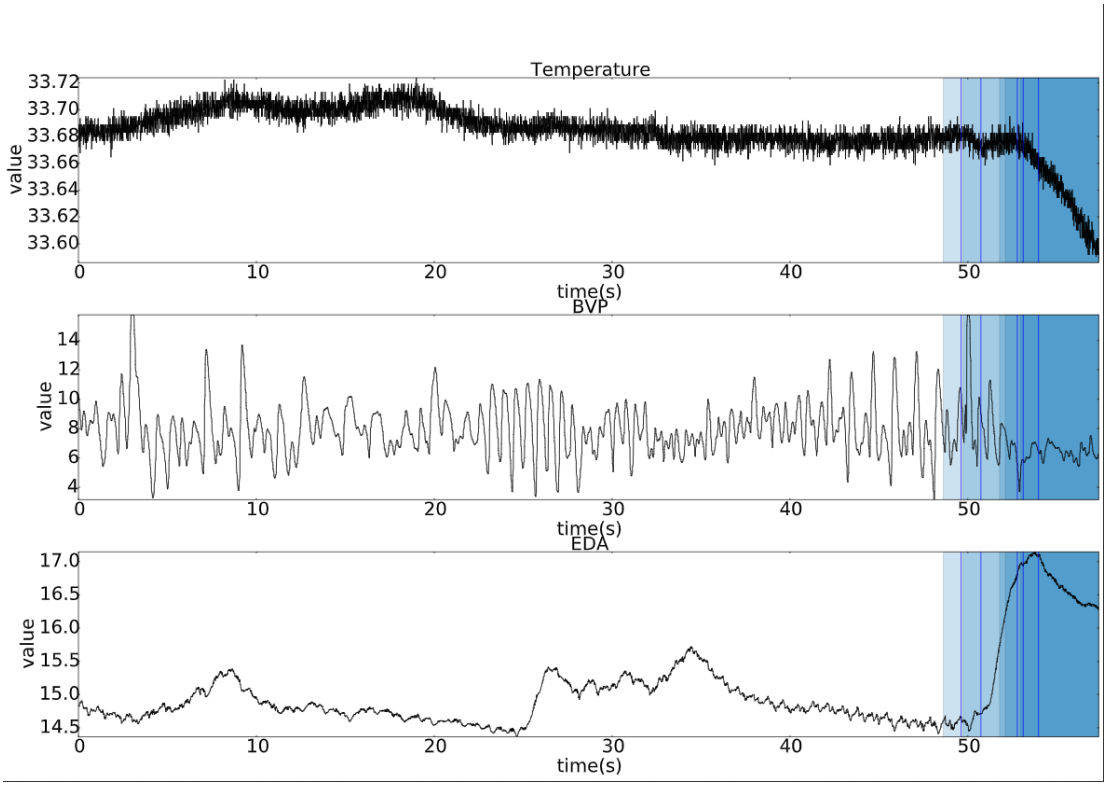


Figure 5.1: Example time series with fearful events (spider attack in mountain level)

5.3 Preprocessing

At the beginning there should always be the preparation/preprocessing of the dataset. In previous articles, for example [MÜL+2017b], the most promising events for emotion recognition were examined and the plausibility of the data was also checked. All plots were reviewed manually and all time series with corrupted data were removed from the dataset.

In this thesis much more time series were used and to reduce the manual effort a automatic preprocessing was used.

1. All time series were the participants said about himself/herself in the self-assessment that he/she did not have the emotion we wanted, were automatically moved from the expected emotion into the self-experienced emotion.

2. If the expected emotion, the self-experienced emotion and the observed emotion by the supervisor are all different, the time series was removed.

3. The remaining time series were marked for manually inspection if at least one of the following filter rules applied:

- Values outside the bandwidth of the sensor.
- Values reach maximum or minimum of bandwidth and remain longer than 64 data points in this area.
- Values have changed abruptly with more than 5% of the bandwidth.
- Events are too close together and the time series overlaps.
- A strong acceleration of the left hand was detected.

All filtered data records were manually reviewed after that and then removed if necessary.

After only error-free time series were left, these were converted into a normalized and standardized format. In addition, care has been taken to ensure that the data set is balanced. The dataset was then divided into a 30% training set and a 70% test set.

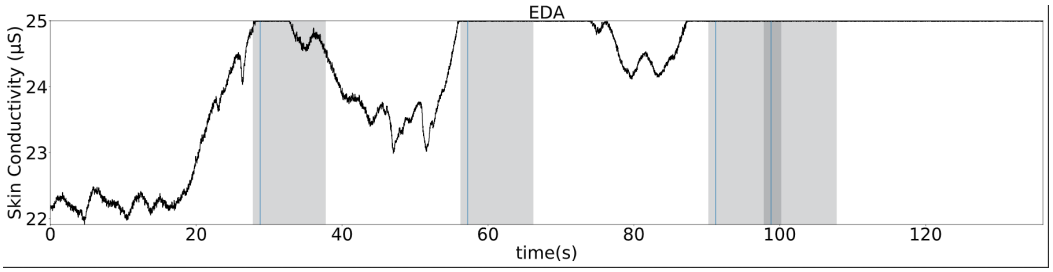


Figure 5.2: Erroneous time series

5.4 Machine Learning Classification

5.4.1 DTW with K-Nearest Neighbor Classification

Over time, many different classification procedures have been developed. However, only a few of them are usable for time series. Dynamic Time Warping (DTW) with K-Nearest Neighbor (kNN) Classification has been used as the standard time series classification method in recent years.

For DTW a cost matrix M x N between the training data M and the test data N is generated. The total costs of the warping path are calculated. This serves as a distance function for the kNN classification.

This method was implemented in Python as a baseline. Subsequently, series of experiments were carried out with different parameters for k, ranging between 1 and 9 (see below).

5.4.2 Feature Extraction with SVM Classification

A feature-based approach was implemented as a second machine learning method. Using tsfresh [4], 1400 features from each time series were extracted, of which x features turned out to be relevant and were selected. Subsequently, SVM was used to classify the features.

5.5 Deep Learning Classification

MLP

The implemented neural network contains three hidden layers with 500 neurons each and an output layer with two neurons. The hidden layers uses ReLU as the activation function and a

[4] https://tsfresh.readthedocs.io/en/latest/index.html

dropout operation with a rate of 0,1 for the input layer, respectively 0,2 - 0,2 - 0,3 for the first, second and third hidden layer.

The whole model uses categorical cross-entropy as the loss function (also known as logarithmic loss). It gives a probability value between zero and one. It is a performance measure where zero loss would be a perfect model.

FCN

The proposed FCN model contains 3 convolution layers and is oriented on the proposed architecture described in [**wang**]. Each layer is a module that contains a base convolution layer, batch normalization and rectifying non-linearity (ReLU). The convolution filters are initialized with the Glorot Scheme [**glorot**], with filter sizes of 8, 5 and 3, for the first, second and third layer respectively. The filter count was fixed to 128 for the first and third convolution layer, as a factor of the time series length. The second layer has 256 filters. The last layers of the FCN, responsilble for class prediction, are a Global Pooling Layer (GAP) and a Softmax layer.

The whole model uses categorical cross-entropy as the loss function and as optimizer we use Adam [**Adam**].

ResNet

The ResNet is built out of 3 identical residual blocks. Every block has 3 convolution layers with filter sizes 8, 5 and 3, all of which are followed by batch normalization and rectifying non-linearity. The shortcut connection is stretched from the block input to its output. Beside this fact, the only difference between a residual block and FCN described above is the filter count, in the current case being 64, 128 and 128, respectively, to allow stacking of blocks with enough complexity. Finally, the last layers of ResNet are the same as in FCN.

5.6 Results

In the following the results for all methods will be presented:

DTW with k Nearest Neighbors

A total of 5 different test runs were carried out with DTW + kNN classification as a baseline for the further experiments. The following k values were used: 1, 3, 5, 7 and 9. As shown in table 5.1, an average accuracy of 70.42% have been achieved. Were the best result was with an

k of 3 and the worst result with a k of 1. And k 3 also had the smallest deviation with $\pm$ 1.21 within the results for each individual sensor.

An interesting finding is, that the results for the EDA sensor are the worst for all sensors in DTW with kNN. Whereby in the manual evaluations with a human eye EDA was always perceived as the best result. ECG was determined as best result per sensor, where it was very difficult to the human eye to detect differences.

The best determined results for each sensor was EDA with $k = 9$, ECG $k = 7$, temperature $k = 3$, respiration $k = 3$ and BVP $k = 3$.

When we would choose always the best k value for each sensor, then we get a mean of 73.3% accuracy with DTW + kNN.

k	Mean	Deviation $\pm$	EDA	ECG	TEMP	RESP	BVP
1	68.59	2.19	66.20	71.40	69.70	65.80	69.70
3	**71.73**	**1.21**	71.40	70.10	**71.10**	**73.70**	**72.40**
5	69.36	3.25	67.50	75.30	65.80	69.70	68.40
7	70.68	3.73	64.90	**76.60**	71.10	71.10	69.70
9	71.72	1.71	**72.70**	74.00	69.70	72.40	69.70
AVG	**70.42**	**2.42**	**68.54**	**73.48**	**69.48**	**70.54**	**69.98**

Table 5.1: DTW with kNN Classifier

Feature Extraction with SVM Classification

The hyper-parameters were determined using grid search, γ between 0.0001 - 1000 and C between 0.0001 - 1000.

In table 5.2 are depicted the five best results. Overall it is a very similar result to DTW with kNN. For example, the results for EDA are also the worst with 69.09%. The best result with 72.79% accuracy was achieved with a γ value of 0.1 and C 0.01.

When we would choose always the best SVM parameters for each sensor, then we get a mean of 73.58% accuracy with Feature extraction + SVM classifier.

Deep Learning Models

The resulting classification accuracy of each deep learning model is shown in Table 5.3. As we can see, there is no clear winner model, that achieves the highest accuracy on every single sensor data.

C	γ	Mean	Deviation $\pm$	EDA	ECG	TEMP	RESP	BVP
0.01	0.01	72.27	5.10	66.20	71.40	72.40	69.70	81.60
0.1	0.1	**72.79**	4.57	68.80	**71.40**	**72.40**	69.70	**81.60**
0.2	0.3	72.52	3.24	70.10	71.40	71.10	71.10	78.90
0.4	0.4	69.89	0.19	70.10	70.10	69.70	69.70	69.70
1	0.8	70.42	0.99	**70.10**	70.10	69.70	**72.40**	69.70
AVG	:	**71.58**	**2.82**	**69.06**	**70.88**	**71.06**	**70.52**	**76.30**

Table 5.2: Feature Extraction with SVM Classifier

	BVP	ECG	EDA	RESP	TEMP	ENSEMBLE
MLP	**83.48**	72.11	71.88	69.37	55.87	75.54
FCN	81.73	75.92	74.78	**75.20**	77.03	**79.97**
ResNet	80.21	75.36	**75.90**	72.64	**78.46**	78.94
LSTM	80.69	**76.10**	75.32	74.43	76.10	78.86
Autoencoder	80.54	74.82	73.71	74.80	75.56	78.40

Table 5.3: Deep Learning Models

The best accuracies came from the BVP sensor data, which means that this sensor can be used to reliably distinguish emotions. The lowest accuracies were observed from the temperature sensor data.

Most importantly the highest accuracy score was delivered by the ensemble of the FCNs. This ensemble uses all the available sensor data and achieves lower accuracies then any model based on the BVP sensor data.

5.7 Discussion of Results

While the results for the BVP sensor with DTW + kNN was rather poor, for Feature extraction + SVM classifier and all deep learning models this sensor reaches the highest accuracy.

As expected, MLP shows the worst accuracies for any sensors.

6 Conclusion

In the context of the master thesis a functional system for the acquisition of the physiological data of the participants in the experimental setup was developed.

The experiments demonstrate the successful use of deep learning models for such non-trivial tasks as emotion recognition in exergames. The proposed models were able to achieve a decent accuracy level despite high data complexity and turned our attention to certain sensors that produce machine distinguishable data. Moreover, we compared the proposed ensembles against each other as well as against single-channel models. The most promising results were obtained with FCN ensemble, leading to the necessity of further research to increase its performance. The lowest accuracy and a tendency for overfitting was seen in MLP ensemble. Finally, each ensemble was capable of beating the single channel models, stressing the necessity to employ ensemble learning for such tasks.

Our future research focus will be to investigate the influence of preprocessing techniques, which may lead to an increasing accuracy. Also, better results may be achieved through the use of similar benchmark datasets containing physiological data such as MAHNOB-HCI. These datasets can be used to train the proposed ensembles, after which the models can be fine-tuned with our own collected dataset.

It was shown that physiological sensors are ready for practice and can be a promising addition or replacement to the existing solutions. This will be achieved by the progressive miniaturization and development of appropriate sensor technology, but also by better algorithms for the recognition of emotions, in particular by the possibilities of deep learning.

6.1 Future work

The evaluations shows important findings for using deep learning models for emotion recognition with multivariate sensor data from our laboratory study. But nevertheless there is room for improvements. Unfortunately, the data set was very small in comparison to data sets that are otherwise used for the training of deep learning models. This quickly resulted in an overfitting and it was very difficult to prevent this by appropriate parameter selection. Much larger test

series would have to be carried out in order to gain a reasonably large data set. In addition, an automated hyper-parameter optimization should be performed [BIB2018].

Another possibility would be to search for other public accessible databases and use them for an initial model training. Afterwards, this pre-trained model can be used with your own data for fine tuning.

If in the end a very good model has emerged, it would have to be integrated into the Emotion-Bike setup to enable real-time analysis of the sensor data.

One of the hardest challenges will be to detect the discrepancy between the time of the event was triggered and the time of when the participant really cognitively apprehended the event in the virtual world. In the manually evaluation of the data we saw often a shift for some second between this two moments.

Bibliography

[AIG2015] Anagnostopoulos, Christos-Nikolaos et al. “Features and classifiers for emotion recognition from speech: a survey from 2000 to 2011”. In: *Artificial Intelligence Review* 43.2 (Feb. 2015), pp. 155–177. ISSN: 1573-7462. DOI: 10.1007/s10462-012-9368-5. URL: https://doi.org/10.1007/s10462-012-9368-5.

[BAG+2017] Bagnall, Anthony et al. “The Great Time Series Classification Bake off: A Review and Experimental Evaluation of Recent Algorithmic Advances”. In: *Data Min. Knowl. Discov.* 31.3 (May 2017), pp. 606–660. ISSN: 1384-5810. DOI: 10.1007/s10618-016-0483-9.

[BER+2017] Bernin, Arne et al. “Towards More Robust Automatic Facial Expression Recognition in Smart Environments”. In: June 2017, pp. 37–44. DOI: 10.1145/3056540.3056546.

[BIB2018] Bibaeva, Victoria. “Using Metaheuristics for Hyper-Parameter Optimization of Convolutional Neural Networks”. In: *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)* (2018), pp. 1–6.

[BRE2001] Breiman, Leo. “Random Forests”. In: *Mach. Learn.* 45.1 (Oct. 2001), pp. 5–32. ISSN: 0885-6125. URL: https://doi.org/10.1023/A:1010933404324.

[CKF2016] Christ, Maximilian et al. “Distributed and parallel time series feature extraction for industrial big data applications”. In: (Oct. 2016).

[CLH2009] Cai, J. et al. “The Research on Emotion Recognition from ECG Signal”. In: *2009 International Conference on Information Technology and Computer Science*. Vol. 1. July 2009, pp. 497–500. DOI: 10.1109/ITCS.2009.108.

[COR+2016] Corneanu, C. A. et al. “Survey on RGB, 3D, Thermal, and Multimodal Approaches for Facial Expression Recognition: History, Trends, and Affect-Related Applications”. In: *IEEE Transactions on Pattern Analysis and Ma-*

chine Intelligence 38.8 (Aug. 2016), pp. 1548–1568. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2016.2515606.

[CSB2011] Cowie, Roddy et al. "Emotion: Concepts and Definitions". In: Oct. 2011, pp. 9–30. DOI: 10.1007/978-3-642-15184-2_2.

[CTA2015] Chen, Jinhui et al. "Facial expression recognition with multithreaded cascade of rotation-invariant HOG". In: (Sept. 2015), pp. 636–642. DOI: 10.1109/ACII.2015.7344636.

[CUH2015] Clevert, Djork-Arné et al. "Fast and accurate deep network learning by exponential linear units (elus)". In: *arXiv preprint arXiv:1511.07289* (2015).

[EF1978] Ekman, P. and W. Friesen. *Facial Action Coding System: A Technique for the Measurement of Facial Movement.* Palo Alto: Consulting Psychologists Press, 1978.

[EFE1972] Ekman, P. et al. *Emotion in the human face: guide-lines for research and an integration of findings*. Pergamon general psychology series. Pergamon Press, 1972. ISBN: 978-0-08-016643-8. URL: https://doi.org/10.1016/B978-0-08-016643-8.50001-X.

[F+2015] Feraru, Silvia Monica, Dagmar Schuller, et al. "Cross-language acoustic emotion recognition: An overview and some tendencies". In: *2015 International Conference on Affective Computing and Intelligent Interaction (ACII)*. IEEE. 2015, pp. 125–131.

[FAN+2016] Fan, Yin et al. "Video-based Emotion Recognition Using CNN-RNN and C3D Hybrid Networks". In: *Proceedings of the 18th ACM International Conference on Multimodal Interaction*. ICMI '16. Tokyo, Japan: ACM, 2016, pp. 445–450. ISBN: 978-1-4503-4556-9. DOI: 10.1145/2993148.2997632. URL: http://doi.acm.org/10.1145/2993148.2997632.

[FS1995] Freund, Yoav and Robert E. Schapire. "A Decision-theoretic Generalization of On-line Learning and an Application to Boosting". In: EuroCOLT '95 (1995), pp. 23–37. URL: http://dl.acm.org/citation.cfm?id=646943.712093.

[GAH2017] Gogate, M. et al. "Deep learning driven multimodal fusion for automated deception detection". In: *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. Nov. 2017, pp. 1–6. DOI: 10.1109/SSCI.2017.8285382.

[HAA+2004] Haag, Andreas et al. “Emotion Recognition Using Bio-sensors: First Steps towards an Automatic System”. In: *Affective Dialogue Systems*. Ed. by Elisabeth André et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 36–48. ISBN: 978-3-540-24842-2.

[HAY+2009] Haykin, Simon S et al. *Neural networks and learning machines*. Vol. 3. Pearson education Upper Saddle River, 2009.

[HE+2016] He, Kaiming et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[HOL2009] Holodynski, Manfred. “Milestones and Mechanisms of Emotional Development”. In: *Emotions as Bio-cultural Processes*. Ed. by Hans J. Markowitsch and Birgitt Röttger-Rössler. New York, NY: Springer US, 2009, pp. 139–163. ISBN: 978-0-387-09546-2. DOI: 10.1007/978-0-387-09546-2_7. URL: https://doi.org/10.1007/978-0-387-09546-2_7.

[HOR2015] Hornschuh, Jonas. “Weiterentwicklung eines Fahrradergometers als intuitive Steuerung für virtuelle Welten”. In: *Bachelorarbeiten HAW* (2015). URL: https://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/bachelor/hornschuh.pdf.

[HS1997] Hochreiter, Sepp and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[IS2015] Ioffe, Sergey and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *CoRR* abs/1502.03167 (2015). arXiv: 1502.03167. URL: http://arxiv.org/abs/1502.03167.

[JS2001] Juslin, P.N. and J.A. Sloboda. *Music and Emotion: Theory and Research*. Series in affective science. Oxford University Press, 2001. ISBN: 9780192631886. URL: https://books.google.de/books?id=5FfvnQEACAAJ.

[KA2009] Kim, Jonghwa and Elisabeth Andre. “Emotion Recognition Based on Physiological Changes in Music Listening”. In: *IEEE transactions on pattern analysis and machine intelligence* 30 (Jan. 2009), pp. 2067–83. DOI: 10.1109/TPAMI.2008.26.

[KAM+2018] Kamenz, A. et al. “Classification of Physiological Data in Affective Exergames”. In: *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*. Nov. 2018, pp. 2076–2081. DOI: 10.1109/SSCI.2018.8628695.

[KAM2016] Kamenz, Andreas. “Emotionserkennung mittels Bio-Sensoren”. In: *Projektberichte HAW* (2016). URL: https://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2016-proj/kamenz.pdf.

[KAM2017] Kamenz, Andreas. “Emotionserkennung mittels Bio-Sensoren - Zeitreihenanalyse”. In: *Projektberichte HAW* (2017). URL: https://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2017-proj/kamenz.pdf.

[KAR+2018] Karim, Fazle et al. “LSTM fully convolutional networks for time series classification”. In: *IEEE Access* 6 (2018), pp. 1662–1669.

[KB2014] Kingma, Diederik and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *International Conference on Learning Representations* (Dec. 2014).

[KK1981] Kleinginna, Paul R. and Anne M. Kleinginna. “A categorized list of emotion definitions, with suggestions for a consensual definition”. In: *Motivation and Emotion* 5.4 (Dec. 1981), pp. 345–379. ISSN: 1573-6644. DOI: 10.1007/BF00992553. URL: https://doi.org/10.1007/BF00992553.

[KK2017] Kletz, F. and J. Kleimann. “Verwendung einer Thermografiekamera im Human-computer interaction Kontext”. In: *Projektberichte HAW* (2017). URL: https://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2017-proj/kletz-kleimann.pdf.

[KOE+2011] Koelstra, Sander et al. “DEAP: A Database for Emotion Analysis Using Physiological Signals”. In: *IEEE Transactions on Affective Computing* 3 (Dec. 2011), pp. 18–31. DOI: 10.1109/T-AFFC.2011.15.

[KRI2014] Krizhevsky, Alex. “One weird trick for parallelizing convolutional neural networks”. In: *CoRR* abs/1404.5997 (2014). URL: http://arxiv.org/abs/1404.5997.

[LBE2015] Lipton, Zachary C et al. “A critical review of recurrent neural networks for sequence learning”. In: *arXiv preprint arXiv:1506.00019* (2015).

[LBH2015] LeCun, Yann et al. “Deep Learning”. In: *Nature* 521 (May 2015), pp. 436–44. DOI: 10.1038/nature14539.

[LCY2013] Lin, Min et al. “Network in network”. In: *arXiv preprint arXiv:1312.4400* (2013).

[LEC+1998] Lecun, Y. et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (Nov. 1998), pp. 2278–2324. ISSN: 0018-9219. DOI: 10.1109/5.726791.

[LEV2003] Levenson, Robert W. “Blood, sweat, and fears: The autonomic architecture of emotion”. In: *Annals of the New York Academy of Sciences* 1000.1 (2003), pp. 348–366.

[MAH+2017] Mahabal, Ashish et al. “Deep-learnt classification of light curves”. In: Nov. 2017, pp. 1–8. DOI: 10.1109/SSCI.2017.8280984.

[MAT2015] Matthiessen, Erik. “Entwicklung einer Gangschaltungs- und Bremsensteuerung für ein Fahrradergometer”. In: *Bachelorarbeiten HAW* (2015). URL: https://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/bachelor/matthiessen.pdf.

[MEH1968] Mehrabian, Albert. “Communication without words”. In: 1968, pp. 53–56.

[MIE2003] Mieny, C.J. *Principles of Surgical Patient*. New Africa Books, 2003, p. 226. ISBN: 9781869280055.

[MÜL+2015] Müller, Larissa et al. “Entertainment Computing - ICEC 2015: 14th International Conference, ICEC 2015, Trondheim, Norway, September 29 - Ocotober 2, 2015, Proceedings”. In: (2015). Ed. by Konstantinos Chorianopoulos et al., pp. 155–168. URL: http://dx.doi.org/10.1007/978-3-319-24589-8_12.

[MÜL+2016] Müller, L. et al. “Physiological data analysis for an emotional provoking exergame”. In: *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. Dec. 2016, pp. 1–8. DOI: 10.1109/SSCI.2016.7850042.

[MÜL+2017a] Müller, L. et al. “Emotional journey for an emotion provoking cycling exergame”. In: *2017 IEEE 4th International Conference on Soft Computing Machine Intelligence (ISCMI)*. Nov. 2017, pp. 104–108. DOI: 10.1109/ISCMI.2017.8279607.

[MÜL+2017b] Müller, L. et al. “Enhancing exercise experience with individual multi-emotion provoking game elements”. In: *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. Nov. 2017, pp. 1–8. DOI: 10.1109/SSCI.2017.8280929.

[MÜL2007] Müller, Meinard. *Information Retrieval for Music and Motion*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007. ISBN: 3540740473.

[MÜL2018] Müller, Larissa. "Affective Computing in Controlled Exergame Environments". In: *Dissertationen HAW* (2018). URL: https://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/phd/mueller.pdf.

[NAM2001] Nanopoulos, Alex et al. "Feature-based Classification of Time-series Data". In: 10 (Jan. 2001), pp. 49–61.

[NK2009] Nieminen, Paavo and Tommi Kärkkäinen. "Ideas about a Regularized MLP Classifier by Means of Weight Decay Stepping". In: *Adaptive and Natural Computing Algorithms, 9th International Conference, ICANNGA 2009, Kuopio, Finland, April 23-25, 2009, Revised Selected Papers*. 2009, pp. 32–41. DOI: 10.1007/978-3-642-04921-7_4. URL: https://doi.org/10.1007/978-3-642-04921-7%5C_4.

[NUM+2014] Nummenmaa, Lauri et al. "Bodily maps of emotions". In: *Proceedings of the National Academy of Sciences* 111.2 (2014), pp. 646–651. ISSN: 0027-8424. DOI: 10.1073/pnas.1321664111. eprint: https://www.pnas.org/content/111/2/646.full.pdf. URL: https://www.pnas.org/content/111/2/646.

[PIC1997] Picard, Rosalind W. *Affective Computing*. Cambridge, MA, USA: MIT Press, 1997. ISBN: 0-262-16170-2.

[PLU1962] Plutchik, Robert. *The Emotions: Facts, Theories, and a New Model*. Studies in psychology, PP24. Random House, 1962. URL: https://books.google.de/books?id=ZMUZAAAAMAAJ.

[PLU2001] Plutchik, Robert. "The Nature of Emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice". In: *American Scientist* 89.4 (2001), pp. 344–350. ISSN: 00030996. URL: http://www.jstor.org/stable/27857503.

[RIN+2015] Ringeval, Fabien et al. "Av+ ec 2015: The first affect recognition challenge bridging across audio, video, and physiological data". In: *Proceedings of the 5th International Workshop on Audio/Visual Emotion Challenge*. ACM. 2015, pp. 3–8.

[RIV+2007] Riva, Giuseppe et al. “Affective Interactions Using Virtual Reality: The Link between Presence and Emotions”. In: *Cyberpsychology behavior : the impact of the Internet, multimedia and virtual reality on behavior and society* 10 (Mar. 2007), pp. 45–56. DOI: 10.1089/cpb.2006.9993.

[RUS1980] Russell, James A. “A circumplex model of affect.” In: *Journal of Personality and Social Psychology* 39.6 (1980), pp. 1161–1178. DOI: 10.1037/h0077714.

[SCH1993] Scherer, Klaus R. “Neuroscience projections to current debates in emotion psychology”. In: *Cognition and Emotion* 7.1 (1993), pp. 1–41. DOI: 10.1080/02699939308409174. eprint: https://doi.org/10.1080/02699939308409174. URL: https://doi.org/10.1080/02699939308409174.

[SCH2007] Schwarz, Norbert. “Cognitive aspects of survey methodology”. In: *Applied Cognitive Psychology* 21.2 (2007), pp. 277–287. DOI: 10.1002/acp.1340. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/acp.1340. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/acp.1340.

[SE1984] Scherer, K.R. and P. Ekman. *Approaches To Emotion, Expression and the nature of emotion*. Taylor & Francis, 1984. ISBN: 9781317757641.

[SIM+1999] Simons, RF et al. “Emotion processing in three systems: the medium and the message.” In: *Psychophysiology* 36.5 (1999), p. 619.

[SLD2016] Shelhamer, Evan et al. “Fully Convolutional Networks for Semantic Segmentation”. In: *PAMI* (2016). URL: http://arxiv.org/abs/1605.06211.

[SPK2018] Serrà, Joan et al. *Towards a universal neural network encoder for time series*. 2018. arXiv: 1805.03908 [cs.LG].

[SRI+2014] Srivastava, Nitish et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *J. Mach. Learn. Res.* 15.1 (Jan. 2014), pp. 1929–1958. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=2627435.2670313.

[SUT+2013] Sutskever, Ilya et al. “On the importance of initialization and momentum in deep learning”. In: *Proceedings of the 30th International Conference on Machine Learning*. Ed. by Sanjoy Dasgupta and David McAllester. Vol. 28. Proceedings of Machine Learning Research 3. Atlanta, Georgia, USA: PMLR, June 2013, pp. 1139–1147. URL: http://proceedings.mlr.press/v28/sutskever13.html.

[TAY+2015] Taylor, Sara et al. “Automatic identification of artifacts in electrodermal activity data”. In: *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE. 2015, pp. 1934–1937.

[WAN+2017] Wang, Jindong et al. *Deep Learning for Sensor-based Activity Recognition: A Survey*. 2017. arXiv: 1707.03502 [cs.CV].

[WLL2015] Wen, G. et al. “An ensemble convolutional echo state networks for facial expression recognition”. In: *2015 International Conference on Affective Computing and Intelligent Interaction (ACII)*. Sept. 2015, pp. 873–878. DOI: 10.1109/ACII.2015.7344677.

[WUN1906] Wundt, W. “Das Gemeingefühl und andere Totalgefühle”. In: *Vorlesungen über die Menschen- und Tierseele.* Verlag von Leopold Voss, Leipzig, 1906, p. 235.

[WYO2017] Wang, Z. et al. “Time series classification from scratch with deep neural networks: A strong baseline”. In: (May 2017), pp. 1578–1585. DOI: 10.1109/IJCNN.2017.7966039.

[YAO+2016] Yao, Anbang et al. “HoloNet: Towards Robust Emotion Recognition in the Wild”. In: *Proceedings of the 18th ACM International Conference on Multimodal Interaction*. ICMI ’16. Tokyo, Japan: ACM, 2016, pp. 472–478. ISBN: 978-1-4503-4556-9. DOI: 10.1145/2993148.2997639. URL: http://doi.acm.org/10.1145/2993148.2997639.

[YMG2016] Yannakakis, Georgios N et al. “Psychophysiology in games”. In: *Emotion in Games*. Springer, 2016, pp. 119–137.

[ZAG2017] Zagaria, Sebastian. “Emotional adäquat reagierende KI-Agenten zur Erhöhung von Immersion in Computerspielen”. In: *Masterarbeiten HAW* (2017). URL: https://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/zagaria.pdf.

[ZAH+2018] Zahangir Alom, Md et al. “The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches”. In: *arXiv preprint arXiv:1803.01164* (2018).

[ZHA+2017] Zhao, B. et al. “Convolutional neural networks for time series classification”. In: *Journal of Systems Engineering and Electronics* 28.1 (Feb. 2017), pp. 162–169. DOI: 10.21629/JSEE.2017.01.18.

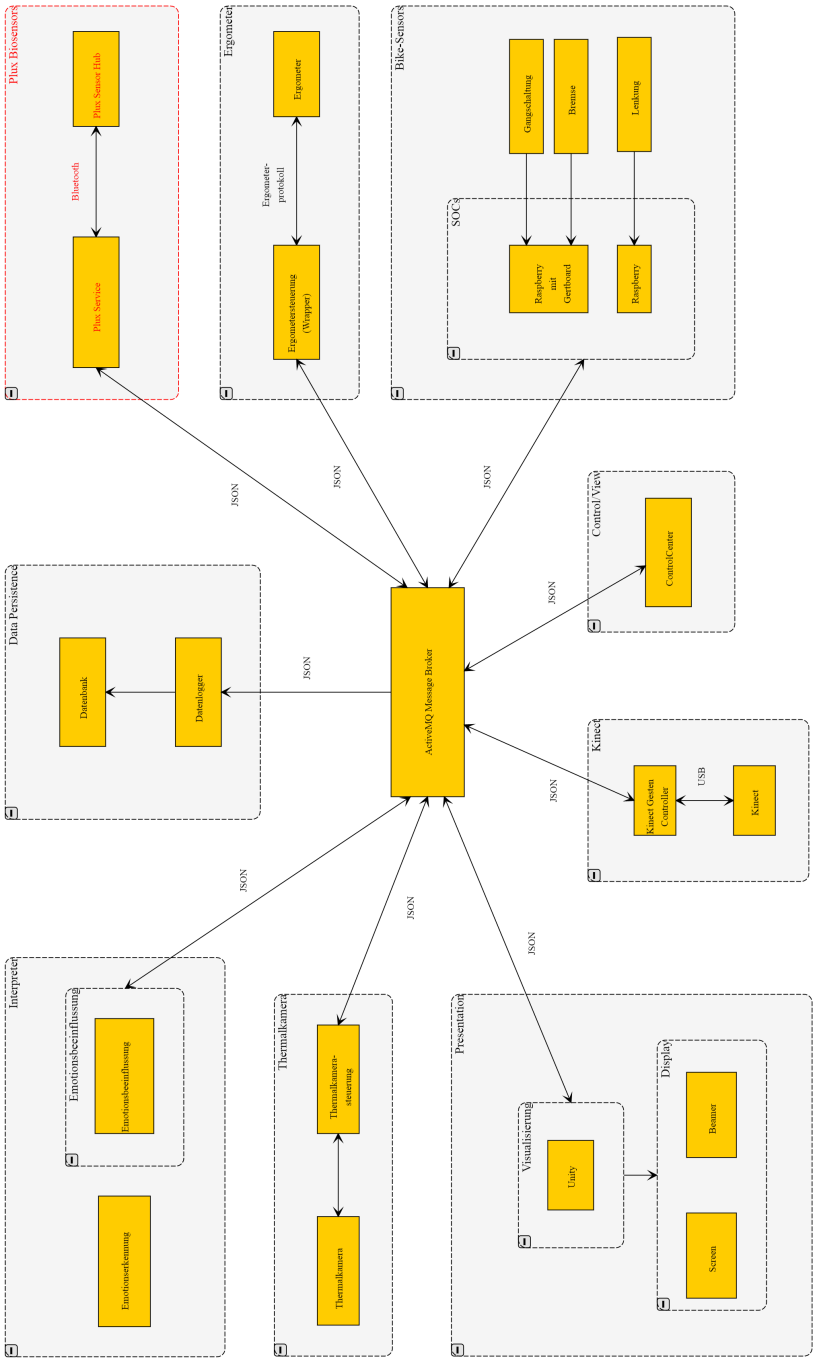
[ZHO+2017] Zhong, B. et al. “Emotion recognition with facial expressions and physiological signals”. In: *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. Nov. 2017, pp. 1–8. DOI: 10.1109/SSCI.2017.8285365.

List of Tables

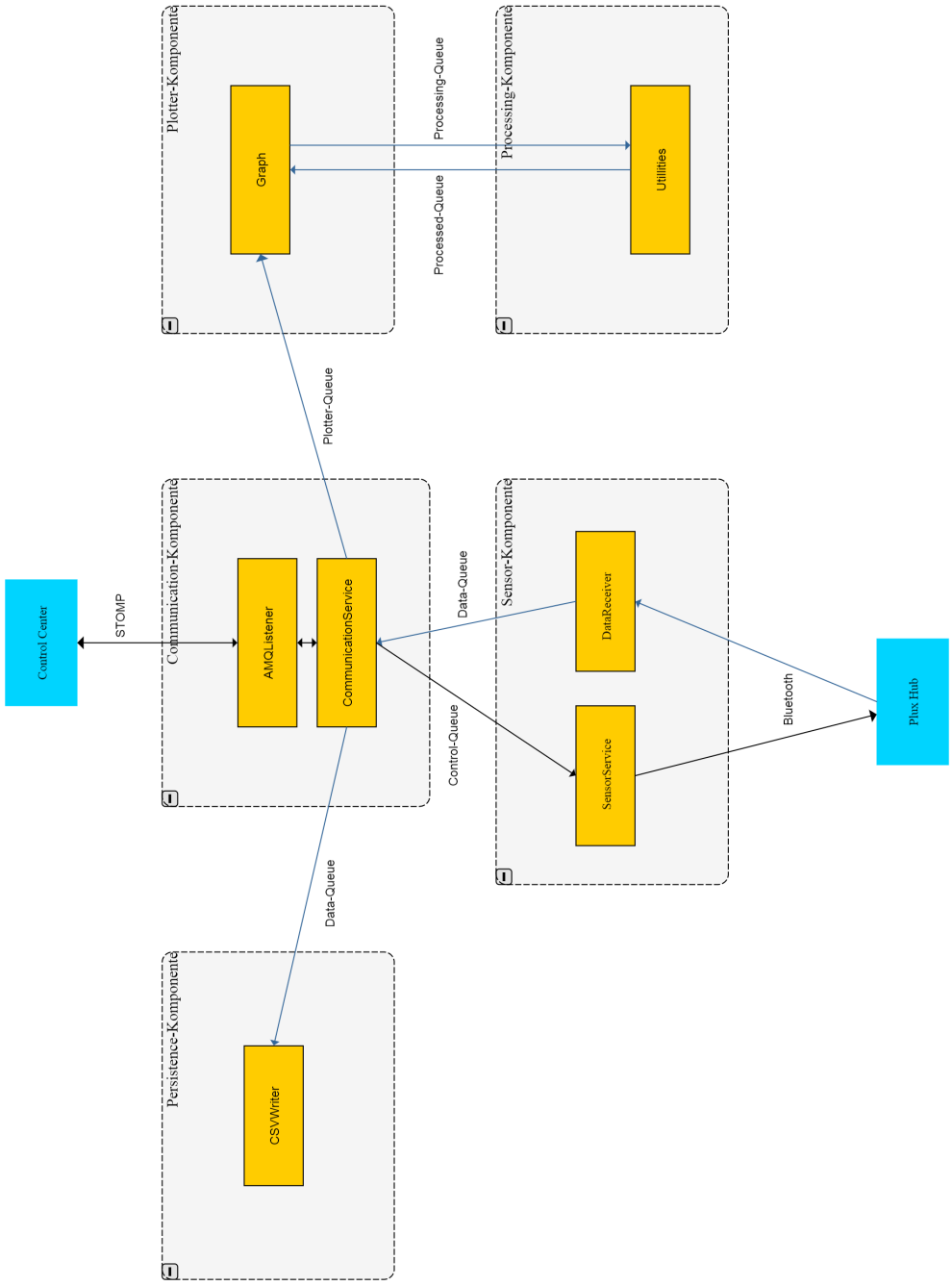
List of Figures

Appendices

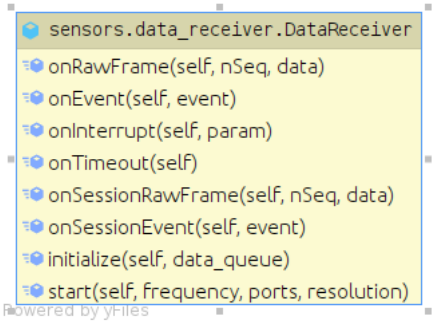
EmotionBike Component Diagram

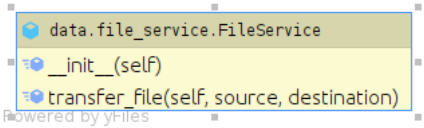


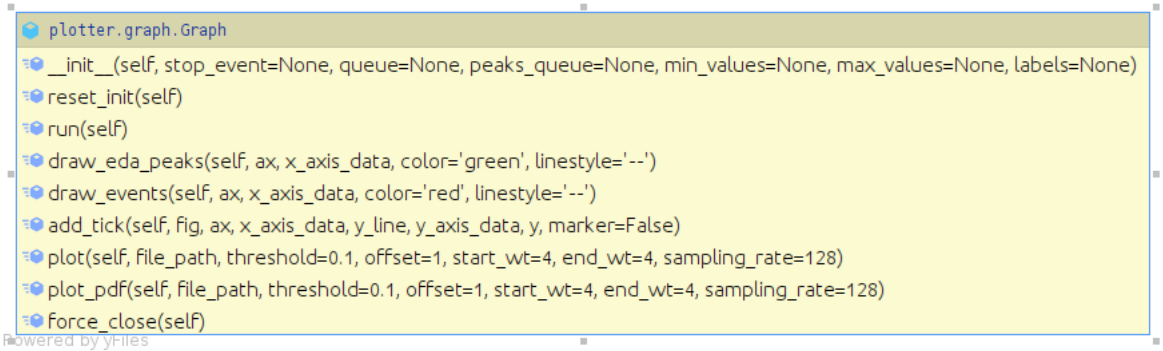
Sensor Service Component Diagram

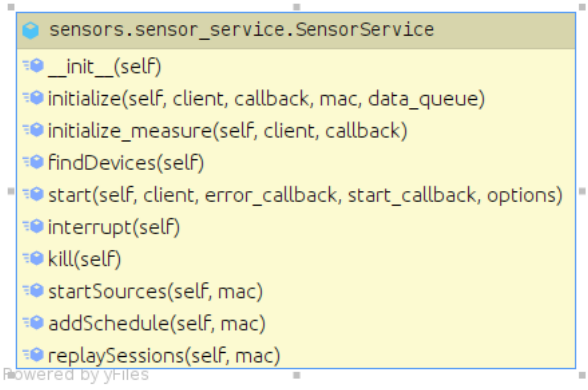


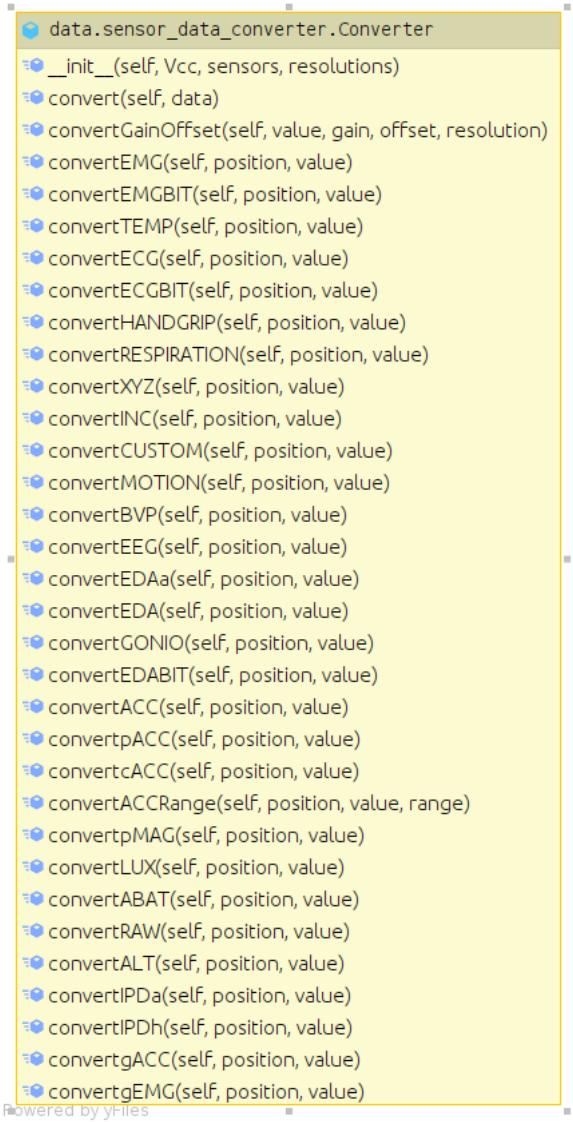
Sensor Service component diagram


sensors.data_receiver.DataReceiver
onRawFrame(self, nSeq, data)
onEvent(self, event)
onInterrupt(self, param)
onTimeout(self)
onSessionRawFrame(self, nSeq, data)
onSessionEvent(self, event)
initialize(self, data_queue)
start(self, frequency, ports, resolution)
Powered by yFiles

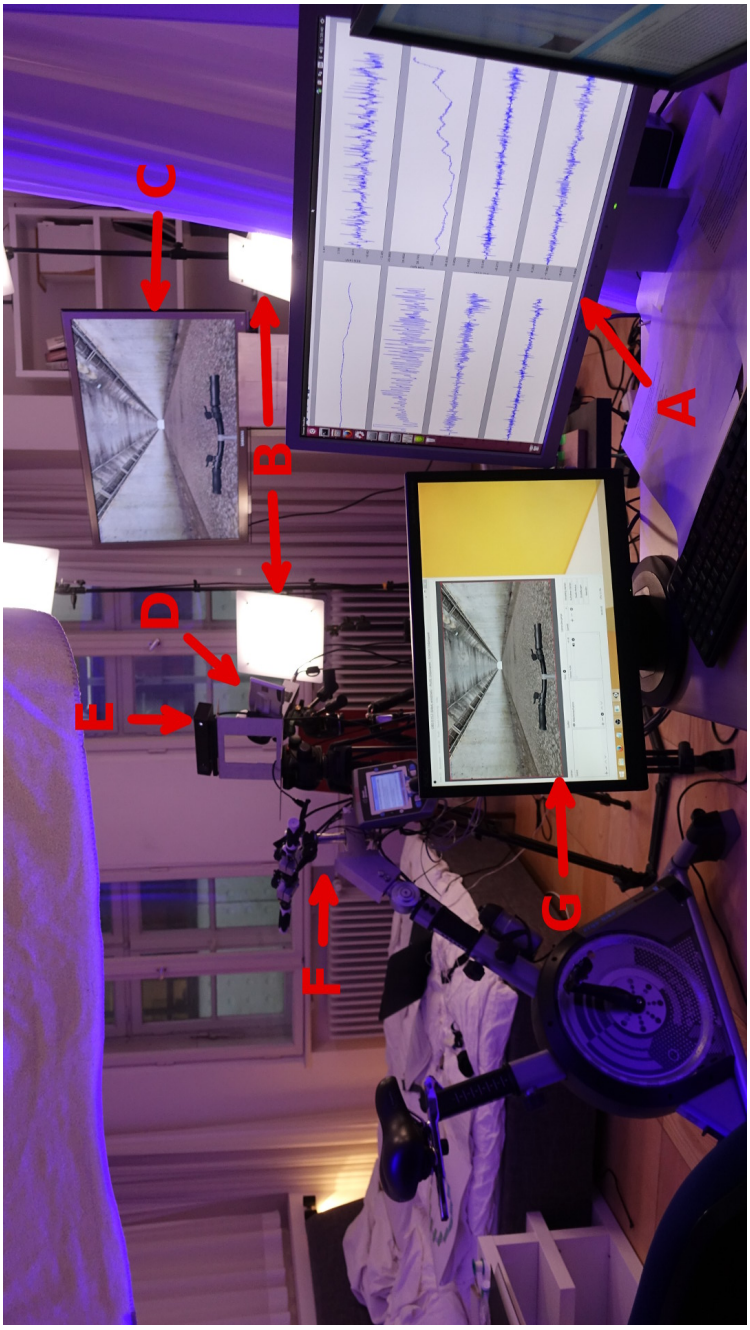

data.file_service.FileService
__init__(self)
transfer_file(self, source, destination)
Powered by yFiles


plotter.graph.Graph
__init__(self, stop_event=None, queue=None, peaks_queue=None, min_values=None, max_values=None, labels=None)
reset_init(self)
run(self)
draw_eda_peaks(self, ax, x_axis_data, color='green', linestyle='--')
draw_events(self, ax, x_axis_data, color='red', linestyle='--')
add_tick(self, fig, ax, x_axis_data, y_line, y_axis_data, y, marker=False)
plot(self, file_path, threshold=0.1, offset=1, start_wt=4, end_wt=4, sampling_rate=128)
plot_pdf(self, file_path, threshold=0.1, offset=1, start_wt=4, end_wt=4, sampling_rate=128)
force_close(self)
Powered by yFiles


sensors.sensor_service.SensorService
__init__(self)
initialize(self, client, callback, mac, data_queue)
initialize_measure(self, client, callback)
findDevices(self)
start(self, client, error_callback, start_callback, options)
interrupt(self)
kill(self)
startSources(self, mac)
addSchedule(self, mac)
replaySessions(self, mac)
Powered by yFiles


data.sensor_data_converter.Converter
__init__(self, Vcc, sensors, resolutions)
convert(self, data)
convertGainOffset(self, value, gain, offset, resolution)
convertEMG(self, position, value)
convertEMGBIT(self, position, value)
convertTEMP(self, position, value)
convertECG(self, position, value)
convertECGBIT(self, position, value)
convertHANDGRIP(self, position, value)
convertRESPIRATION(self, position, value)
convertXYZ(self, position, value)
convertINC(self, position, value)
convertCUSTOM(self, position, value)
convertMOTION(self, position, value)
convertBVP(self, position, value)
convertEEG(self, position, value)
convertEDAa(self, position, value)
convertEDA(self, position, value)
convertGONIO(self, position, value)
convertEDABIT(self, position, value)
convertACC(self, position, value)
convertpACC(self, position, value)
convertcACC(self, position, value)
convertACCRange(self, position, value, range)
convertpMAG(self, position, value)
convertLUX(self, position, value)
convertABAT(self, position, value)
convertRAW(self, position, value)
convertALT(self, position, value)
convertIPDa(self, position, value)
convertIPDh(self, position, value)
convertgACC(self, position, value)
convertgEMG(self, position, value)
Powered by yFiles

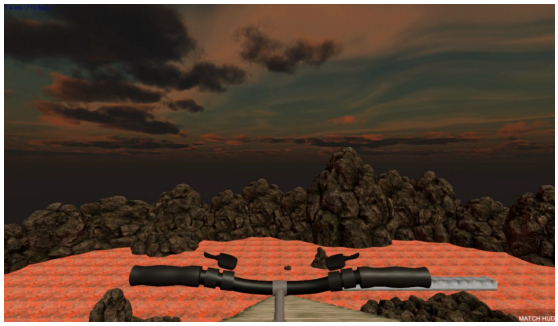
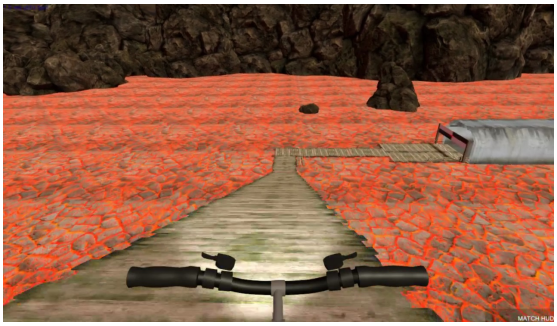
Emotion Bike Setup



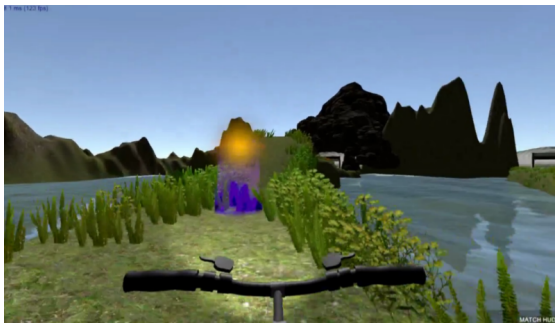
A: Real-time preview of biosensor data, B: Lightning, C: Game flatscreen, D: Thermo camera, E: Kinect camera, F: Ergometer with handlebar & brakes, G: Game control

Levels

Downhill

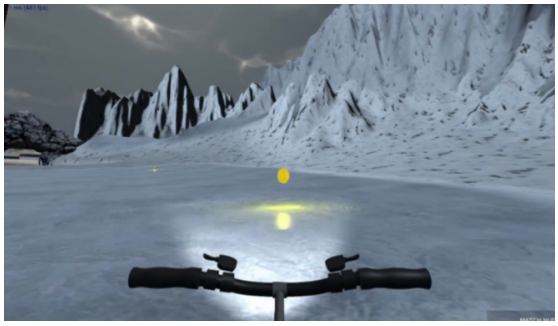
Challenge




Cliff


Explosion


Frozen Sea

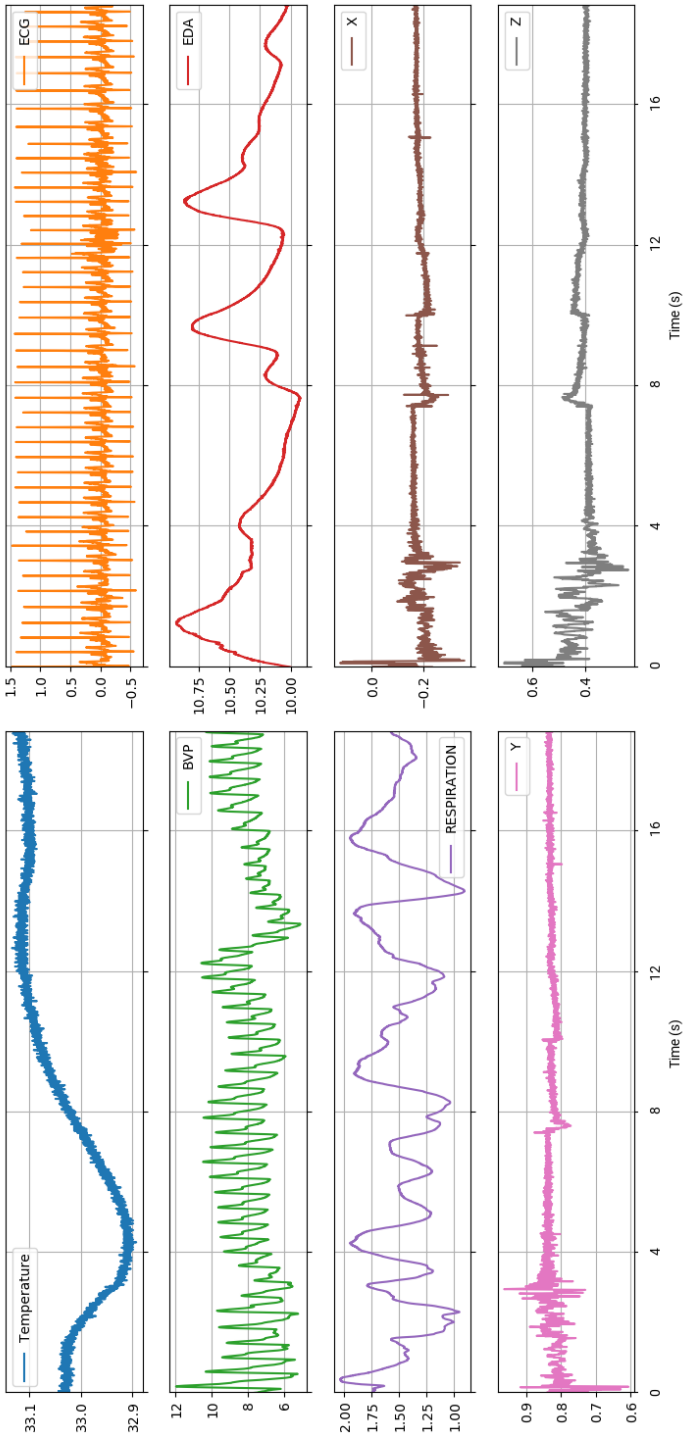


Glade

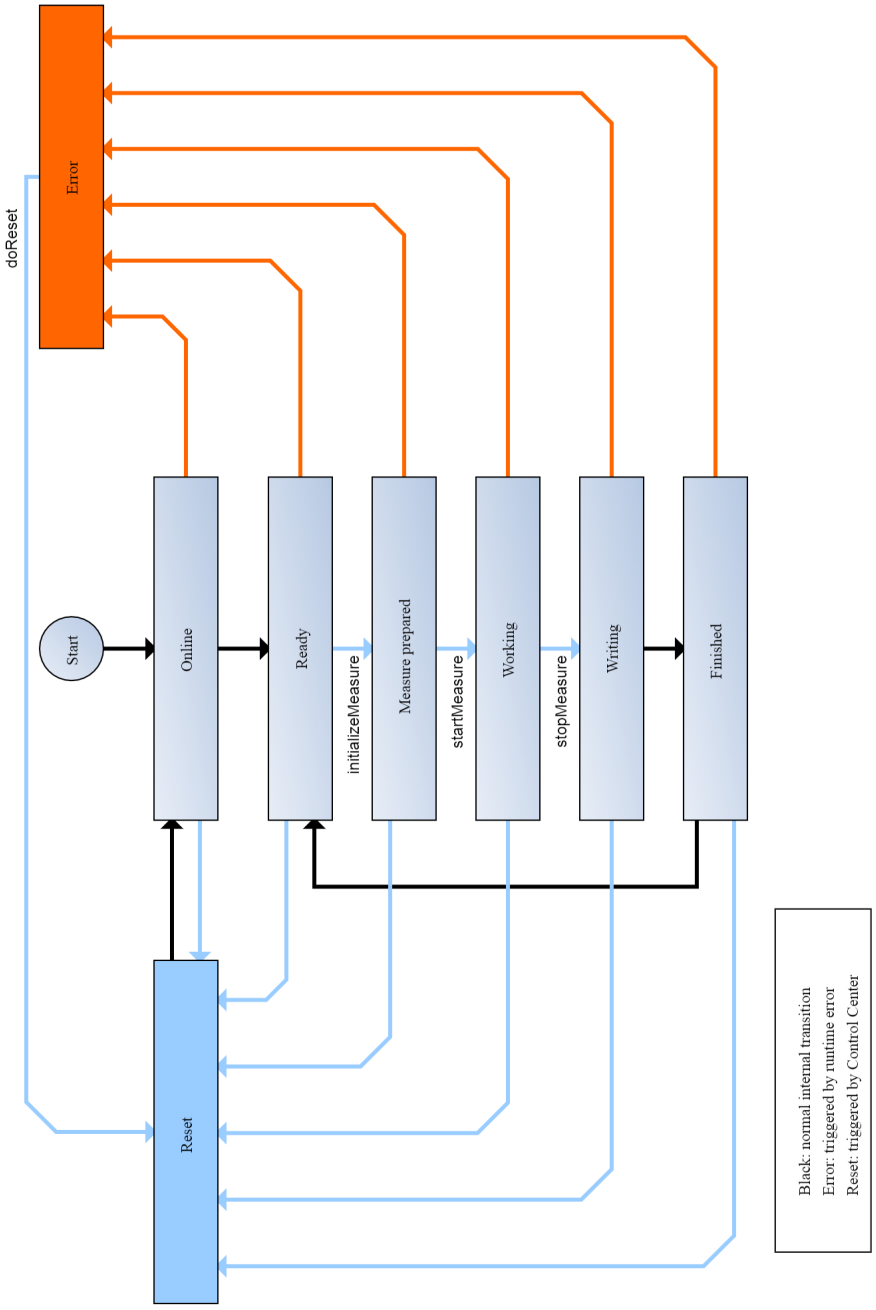
Mountain

Treehouse

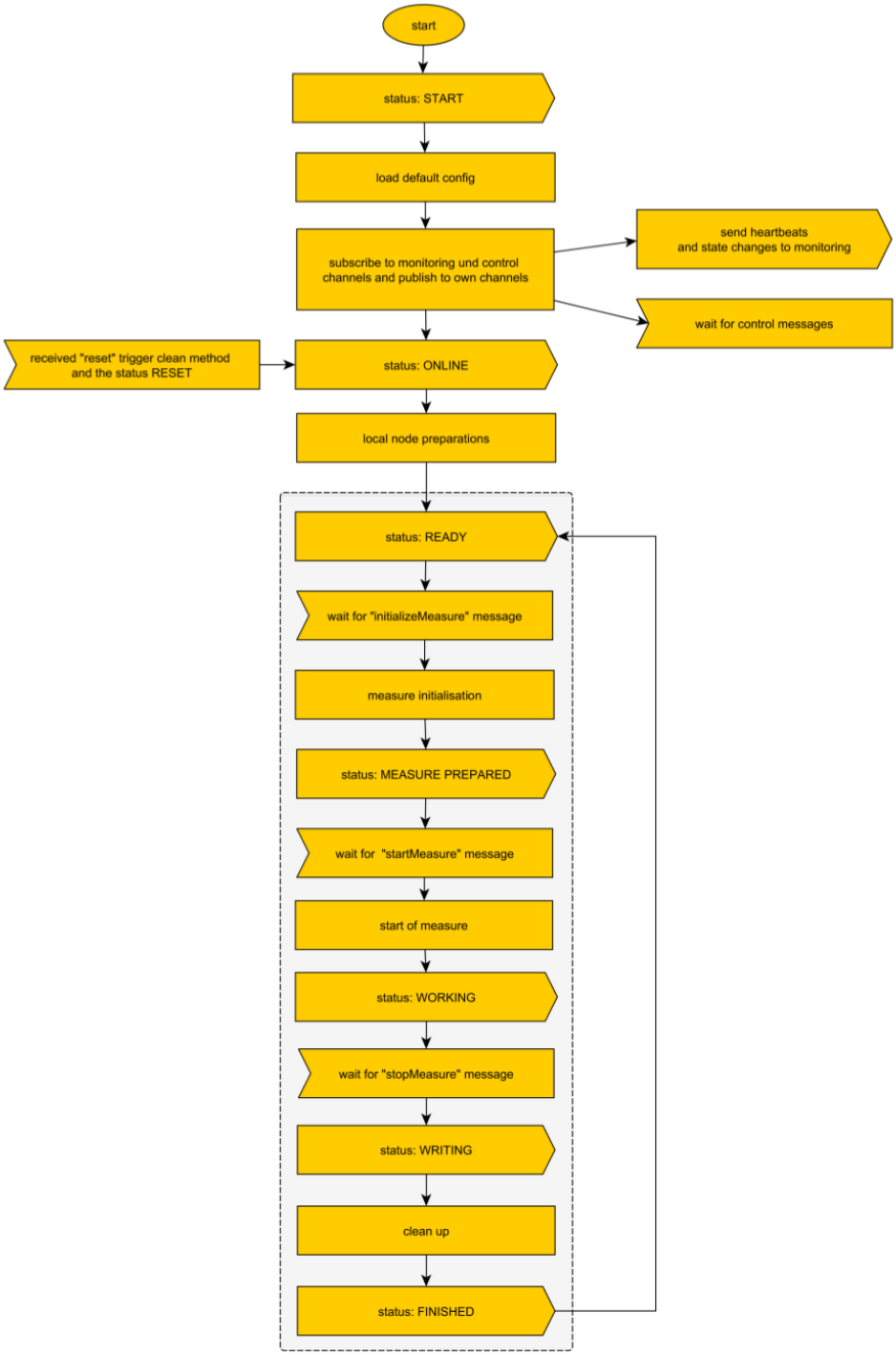
Sensor Diagrams



Node State Machine



Node Full Life Cycle



Plux Sensor Characteristics

Electro Dermal Activity (EDA)

- Measuring range: 0-13 S
- Bandwidth: 0 - 3 Hz
- Impedance: >1 GOhm
- Common-Mode Rejection (CMR): 100 dB

Electrocardiography (ECG)

- Gain: 1000
- Measuring range: $\pm$1.5 mV (at VCC = 3 V)
- Bandwidth: 0.5 - 100 Hz
- Impedance: >100 GOhm
- Common-Mode Rejection (CMR): 100 dB

Blood Volume Pulse (BVG)

- Gain: 34
- Wave length: 670 nm
- Bandwidth: 0.02 - 2.1 Hz

Respiration

- Type: Piezoelectric film

Temperature

- Type: NTC Thermistor
- Measuring range: 0 - 50 °C

Accelerometer

- Type: MEMS
- Axes: 3
- Measuring range: ±3,6G
- Bandwidth: 0 - 50Hz