DTU Compute
Department of Applied Mathematics and Computer Science

Enhancing Deep Learning Performance with Simulated Physiological Data for Acute Stress Prediction

Master's Thesis

Enhancing Deep Learning Performance with Simulated Physiological Data for Acute Stress Prediction

Master's Thesis
January 31st, 2024

By
Harald Vilhelm Skat-Rørdam, Mia Hang Knudsen, & Simon Nørby Knudsen

With supervisors
Line Katrine Harder Clemmensen, lkhc@dtu.dk
Sneha Das, sned@dtu.dk
Nicole Nadine Lønfeldt, nicole.nadine.loenfeldt@regionh.dk

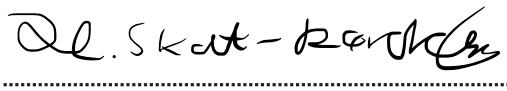
Cover photo: Vibeke Hempler, 2012
Published by: DTU, Department of Applied Mathematics and Computer Science, Richard Petersens Plads, Building 324, 2800 Kgs. Lyngby Denmark
www.compute.dtu.dk

Approval

This thesis has been prepared over five months at the Department of Applied Mathematics and Computer Science, at the Technical University of Denmark, DTU, in partial fulfillment for the degree Master of Science in Engineering, MSc Eng.

It is assumed that the reader has a basic knowledge of the areas of mathematics, statistics, and artificial intelligence.

Harald Vilhelm Skat-Rørdam - s175393

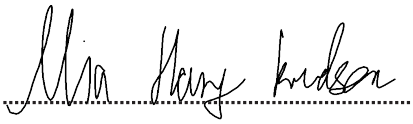


Signature

31/jan 2024

Date

Mia Hang Knudsen - s183998

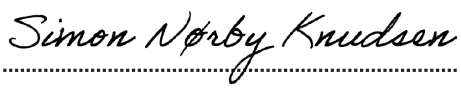


Signature

31/jan 2024

Date

Simon Nørby Knudsen - s174479



Signature

31/jan 2024

Date

Abstract

The area of mental health and disabilities, such as obsessive-compulsive disorder (OCD), has received increased attention due to a large population, particularly adolescents, who suffer from the disorder. Through the application of machine learning, this project aims to support children with OCD in their daily lives, by providing timely warnings against OCD episodes. In the medical field, data scarcity is a common obstacle, and few studies have collected physiological data in the wild, especially concerning OCD in adolescents. However, a study called Wrist Angel, conducted at Børne- og Ungdomspsykiatrisk (Child and Adolescent Psychiatry) Center, Gentofte Hospital collected 8 weeks of daily physiological data from adolescents diagnosed with OCD. It showed that it is feasible to classify OCD events using real-world data. This thesis extends the work from the feasibility study by using transfer learning to improve the performance of predicting OCD events. It provides important insights into the efficacy of transfer learning with potential application to other domains.

Using a convolutional neural network (CNN) pre-trained on open-source stress datasets, stress events were predicted based on the physiological data, which are blood volume pulse (BVP), electrodermal activity (EDA), heart rate (HR), and skin temperature (TEMP).

In an exploratory phase, we tested various models, selected the best model architecture for the stress prediction task, and experimented with four datasets. By pre-training models with the best model architecture, we obtained a positive performance, with an F1-score up to 0.95, for predicting acute stress events for open-source datasets. Additionally, we improved performance using simulated data in some cases. However, when applying the pre-trained model to the in-the-wild OCD data from the Wrist Angel study, the model performance was inadequate. This was possibly due to a lot of noise, which is often an issue for in-the-wild time series data. We have nonetheless conducted tests indicating that fine-tuning on random splits with all participants works better than personal fine-tuning and that longer observation windows are not necessarily beneficial for predicting OCD events.

Our study contributes to the foundation for a new way of utilizing transfer learning to predict OCD events in adolescents based on physiological data.

Acknowledgements

We would like to express our gratitude to the following individuals for their guidance, support, and contributions to the completion of this project:

Line Katrine Harder Clemmensen, *Supervisor*, Applied Mathematics and Computer Science, Technical University of Denmark.

Sneha Das, *Supervisor*, Applied Mathematics and Computer Science, Technical University of Denmark.

Nicole Nadine Lønfeldt, *External Supervisor*, Child and Adolescent Mental Health Center, Copenhagen University Hospital, Mental Health Services Copenhagen.

To **Anne Katrine Pagsberg**, Department of Clinical Medicine, Faculty of Health and Medical Sciences, University of Copenhagen, Copenhagen, Denmark.
We would also like to express our thanks for the seamless facilitation of the use of computers and for providing an available office at Gentofte Hospital.

Contents

1 List of Abbreviations

Abbreviation	Definition
#	Number of
ACC	Acceleration
Acc	Accuracy
ADARP	Dataset from the Alcohol and Drug Abuse Research Program
AUC	Area Under the Curve
AUD	Alcohol Use Disorder
Baseline Scenario	In the context of the Wrist Angle dataset results, the baseline scenario is 1-minute data, 0-minute LPT, no activity filter, and pre-trained random
Bi-LSTM	Bidirectional Long Short-Term Memory
BMI	Body Mass Index
BVP	Blood Volume Pulse
CI	Cardiac Index
CNN	Convolutional Neural Network
CNN-LSTM	Convolutional Neural Network, Long Short-Term Memory
DBP	Diastolic Blood Pressure
DTU (dataset)	Dataset used from Technical University of Denmark
ECG	Electrocardiogram
ECGSYN	A dynamical model for generating synthetic electrocardiogram signals
EDA	Electrodermal Activity
EMA	Ecological Momentary Assessment
EMG	Electromyography
FCN	Fully Convolutional Network
FDA	The United States Food and Drug Administration
FDR	False Discovery Rate
FFT	Fast Fourier Transform
FNN	Feedforward Neural Network
GAN	Generative Adversarial Network
HR	Heart Rate
I_{cl}	Clothing insulation rate
IBI	Interbeat Interval
LLM	Large Language Model
LPT	Lead Prediction Time (time between the end of signal and the tag)
LR	Learning Rate
LReLU	Leaky Rectified Linear Unit
LSTM	Long Short-Term Memory
OCD	Obsessive-Compulsive Disorder
PPG	Photoplethysmogram
PT	Pre-Trained
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
ROAD	The AffectiveROAD Dataset
ROC Curve	Receiver Operating Characteristic Curve
SAP	Statistical Analysis Plan
SBP	Systolic Blood Pressure

SCR	Skin Conductance Response
sEMG	Surface Electromyography
T	Train
TEMP	Skin temperature
TSST	Trier Social Stress Test
TTS-CGAN	Transformer Time-Series Conditional GAN
US	United States
Val	Validation
WA	Wrist Angel (dataset)
WEEE	A Multi-Device and Multi-Modal Dataset for Wearable Human Energy Expenditure Estimation
WESAD	Wearable Stress and Affect Detection dataset

2 Introduction

Obsessive-compulsive disorder (OCD) plays a crucial role in the lives of many children and adolescents, as it is a common mental illness that affects approximately 1 to 3% children and adolescents [Walitza et al. 2011]. Furthermore, it is estimated that around 40% of individuals with OCD experience a chronic form of the disease.

There are tools to manage OCD and reduce the severity of symptoms through self-care. These could, for instance, be accepting intrusive thoughts, identifying and resisting, delaying or reducing compulsions, and distracting yourself, etc. [Mind 2023]. But if a warning could be given before the symptoms unfold, then the patient could prepare, and use certain tools ahead of the symptoms to further reduce or even prevent the OCD episode.

In a feasibility study called Wrist Angel [Lønfeldt et al. 2023], in-the-wild data from children and adolescents was collected and analyzed to see whether OCD events could be predicted with machine learning based on physiological data. The conclusion of this study suggests that OCD events can be distinguished from non-OCD events using real-world physiological data. This thesis further develops the work done by Lønfeldt et al., utilizing other methodologies such as deep learning, transfer learning, etc. applied to time series data.

The scope of the study is to evaluate whether transfer learning from stress events is a viable option to improve the predictability of OCD events, which could be generalized to other areas of interest. This is especially relevant for new prediction tasks, where data scarcity poses as a major bottleneck. It is done by assessing whether a pre-trained deep learning model can improve the performance in predicting OCD events compared to traditional machine learning techniques that were previously applied by Lønfeldt et al. We will be using the same dataset, from here on referred to as the Wrist Angel dataset (WA).

The research questions this thesis intends to answer are as follows:

1. What level of performance can we obtain by fine-tuning a pre-trained stress prediction model?
2. To what extent can this pre-trained model perform well in predicting OCD events?
3. How do different factors affect the performance in predicting OCD events?

The structure of this thesis is visualized in fig. 2.1. The current introduction follows a literature review of previous studies on models using physiological data to predict stress. Then comes a data section that describes the different physiological data of the Empatica E4 wristband [Empatica 2023c]; the Wrist Angel data, open source stress data, and simulated data. The next chapter outlines the different methods and theories used throughout the project. The thesis will then report and analyze results from the performance of various models on open-source datasets and the results of applying our final model on the Wrist Angel dataset. Hereafter, we discuss these results, review why we see the results we have, and what we can learn from them. Finally, we will conclude the project and put our thesis into perspective by suggesting future work.

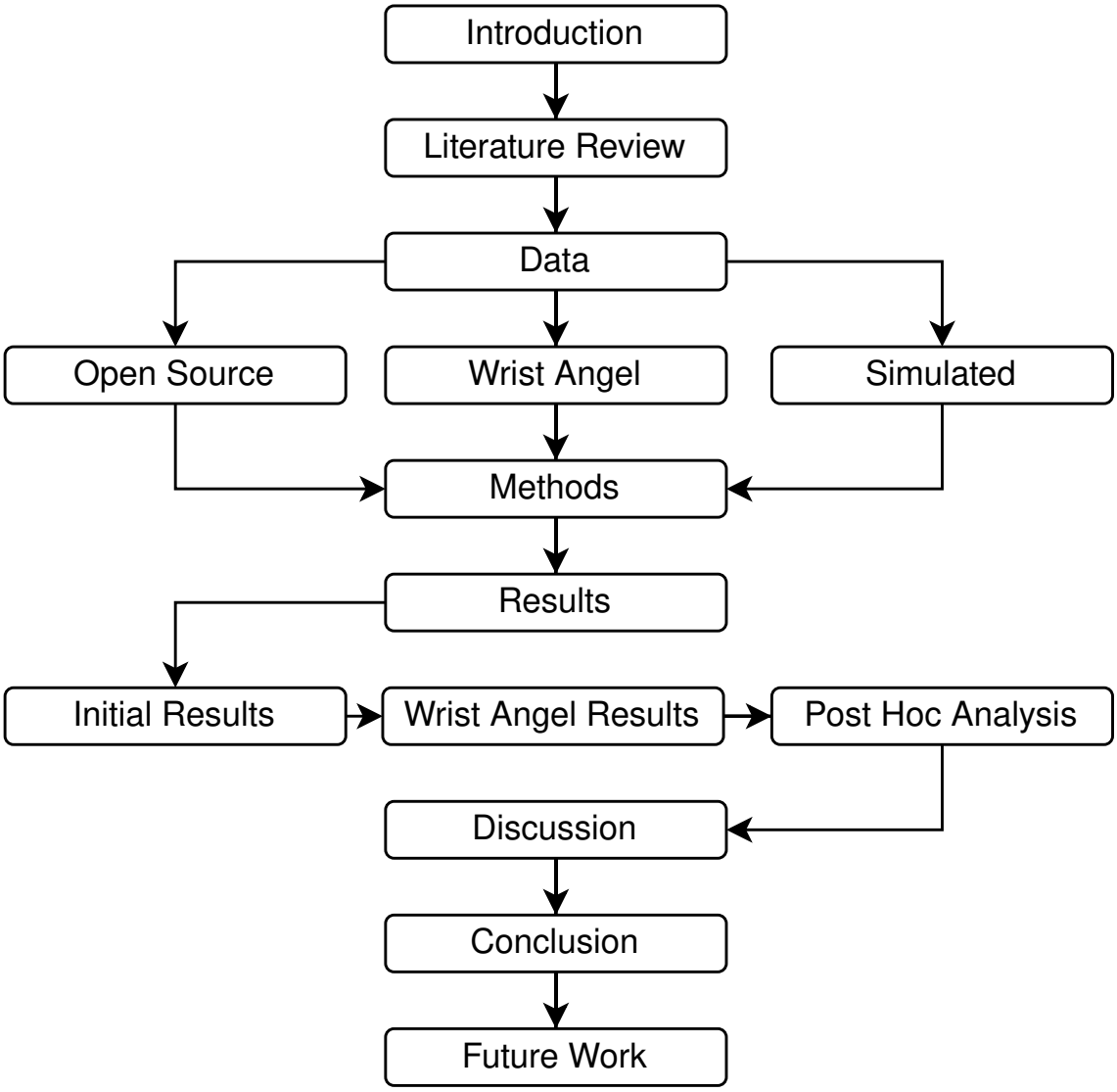


Figure 2.1: Structure of the thesis.

3 Literature Review

The following review will examine various sources of literature. The first section will be on the background of the study and focus on the correlation between physiological signals, acute stress, and OCD. Then followed by relevant literature, which revolves around the possibility of predicting OCD events, including the Wrist Angel study, on which this thesis is based on. Hereafter, the literature for simulating data will be summarized and deep learning models and methods will be examined within the field of event prediction from physiological signals.

3.1 Background

At times, everyone experiences episodes of stress, as well as anxiety caused by various forms of mental illness. These episodes can be characterized as acute stress [Robinson 1990], and repeated episodes can be unpleasant and unhealthy [Scott 2021].

Giannakakis et al. reviewed multiple studies to create an overview of the effects of stress on different physiological signals. They presented tables of multiple physiological signals and different studies' findings of significant differences in the physiological signal during stress. Among others, they state that "Heart rate (HR) is the most prominent feature which increases significantly during stress"[Giannakakis et al. 2022]. Furthermore, they present multiple studies showing a significant increase in Blood Volume Pulse (BVP) and Electrodermal Activity (EDA) during stress. For skin temperature (TEMP), there are not as many studies that agree on the same significant difference during stress. Some studies have found significant differences in TEMP, but it's worth noticing that it seems that TEMP either increases or decreases depending on the part of the body that it was recorded from. Generally, their review suggests that the four signals with which we are working (HR, BVP, EDA, and TEMP) are affected by stress.

According to the National Health Service of England, OCD consists of 3 main elements: obsessions, emotions, and compulsions. Intrusive thoughts cause an intense feeling of anxiety or distress which leads to repetitive behaviors [*Symptoms - Obsessive compulsive disorder (OCD)* 2021].

Physiological signals indicate that stress is relevant for OCD events, as these events are often characterized by anxiety [Abramowitz, Wheaton, and Storch 2008], which is interrelated to stress [Robinson 1990]. Therefore, in the physiological signal, it should be possible to detect if a patient feels stressed due to their OCD symptoms.

3.2 Wrist Angel Study

In the field of OCD, a proof-of-concept experiment was conducted, namely the Wrist Angel Feasibility study [Lønfeldt et al. 2023; Olesen, Das, et al. 2023]. Lønfeldt et al.; Olesen, Das, et al. collected 8 weeks of physiological data from adolescents diagnosed with OCD. They divided the data into 5-minute windows and utilized feature extraction and simple machine learning algorithms such as random forest. With this approach, OCD event prediction based on in-the-wild physiological data was proven to be feasible. Their findings seen in [Lønfeldt et al. 2023] include, but are not limited to the following: Better performance for generalized models compared to personal models. The best performance was obtained by tree-based models with up to 70% accuracy. Lastly, the BVP features were

the most important. Lønfeldt et al. mentions transfer learning and using signals as time series observations as possible improvements.

With the increase in available temporal data [Längkvist, Karlsson, and Loutfi 2014], the potential to use models to process these data has also increased. However, the complexity of real-world data also increases in comparison to data that can be described with equations. According to [Yang and Wu 2006] mining sequence and time series data is considered to be among the top 10 most challenging problems in data mining, due to the contamination of noise in the data. Data with temporal components come from numerous fields of life illustrated by Längkvist, Karlsson, and Loutfi, where they evaluated deep learning and unsupervised feature learning for various data sets such as physiological data, stock prediction, speech recognition, etc. In our analysis of the Wrist Angel data, we aim to model the data as a time series with an autoencoder. The idea is that we, with this approach can retain all information in the data, and hopefully use the information to build a better prediction model.

The autoencoder may prove to be beneficial for the field of health care and mental health, allowing a more empirical evaluation of the patient's status. The autoencoder will not need labeled data, which means they can learn in an unsupervised manner, thus more data can be used to train models to capture patterns within data. As stated in [Längkvist, Karlsson, and Loutfi 2014] labeling medical data is expensive due to the size of the data and the requirements of professionals to hand label the data, thus using feature learning algorithms is advantageous.

3.3 Data Simulation

Open-source data within the field of physiological signals can be scarce; therefore, using simulated/synthetic data is one way of combating this issue. In other domains such as image analysis, using simulated data has proven to improve the performance of models. [Tremblay et al. 2018] found that using training data only created by synthetic images using domain randomization and thereafter fine-tuning with real-life images, has proven to produce better results than a model trained only on real-world data. [F. Wang et al. 2019] have demonstrated training a model only on simulated data to be feasible. Here, the simulations were done by closely imitating the data collection process. However, two studies [Minor et al. 2020; Alkhalifah, H. Wang, and Ovcharenko 2022] have both found that using raw simulated data without any augmentation had significantly worse performance compared to using an augmented version.

We wanted to simulate signals similar to the data collected by the E4 wristband [*E4 sensors* 2023]: HR, EDA, BVP, and TEMP. It is possible to simulate the data with two different methods, either based on physiological models. Alternatively, a model-based simulation can simulate data and imitate any type of real data.

Physiological Simulation

To simulate the four physiological signals from the E4 wristband two Python packages were used, Neurokit2 [Makowski et al. 2021] and JOS-3 [Takahashi et al. 2021]. Neurokit2 is a Python package that can be used for physiological processing. In addition to being able to analyze physiological data, finding peaks, clean data, etc., it can also generate physiological data, which is important for our use case. It has direct support in generating ECG (electrocardiography), PPG (photoplethysmogram), respiration, EDA (electrodermal activity), and EMG (electromyography) data. With these signal generation pipelines, each of the physiological signals from the E4 wristband can be generated (except TEMP, which is generated with JOS-3 as mentioned below). HR can be generated with PPG and ECG, BVP can be generated with PPG, and finally, EDA can be generated

directly. It is important to note that each physiological signal is generated independently, this is contrary to the E4 data wherein the signals are not independent, since they come from the same person. We were unable to find documentation on how the data is generated, i.e., which age and physical traits are being simulated. This is a problem when simulating data for non-adults as the signals are different [Gambera 2021].

For simulating TEMP data, we used the Python package JOS-3 [Takahashi et al. 2021]. The package provides a detailed model, which can be used to simulate thermal physiology data. This is to our knowledge the only model that can be used to simulate human skin temperature. In our case, we are interested in simulating the TEMP of the right wrist. The model provides detailed input parameters to set, such that we can simulate exact situations.

Model-based Simulation
Another possibility of data generation is generative adversarial networks (GAN) specifically transformer-based conditional (TTS-CGAN) as proposed by [Li, Ngu, and Metsis 2022]. The TTS-CGAN model can be trained on existing data and generate synthetic time-series data of arbitrary length. Li, Ngu, and Metsis showed that not only are the generated sequences *"indistinguishable"* from real ones but also that the generated data can help in increasing accuracy in classification tasks when used as additional data[1]. However, a drawback is the amount of data needed to properly simulate the original data. In our case, the computational resources needed were more than we had available, and thus we were unable to generate any data.

3.4 Deep Learning in the Physiological Domain

The task of classifying and labeling physiological signal data has great applicability for complex deep-learning models. [Hu, J. Chen, and Zhou 2022] proposed a novel deep learning model that combines a convolutional neural network (CNN), a transformer-based neural network, and a feedforward neural network (FNN) to classify arrhythmias based on ECG data. With this architecture, they were able to significantly improve performance in different classification tasks.

[Deznabi and Fiterau 2023] created a framework that can handle signals sampled at different frequencies. This is a common issue when dealing with multivariate time series data, and is also the case with data from the E4 wristband. Deznabi and Fiterau proposed a novel framework, MultiWave, which uses wavelets to decompose signals into sub-signals. Each sub-signal is handled by separate components of the model, where each output is combined in a gating mechanism. With this technique, they were able to obtain "top performance in stress and affect detection from wearables"[Deznabi and Fiterau 2023]. Several architectures were implemented including CNN, LSTM, and Transformer-based models.

Additional classification tasks have also been considered. [Kumar et al. 2022] compared the performance of multiple models: Long short-term memory (LSTM) networks, Bidirectional LSTM (Bi-LSTM), attention-based variants of LSTM, convolutional neural network LSTM (CNN-LSTM), and Convolutional-LSTM networks while considering different time windows of 32s and 64s. The task was to predict the respiratory rate from ECG and surface electromyography (sEMG) data. It was found that no single model performed best across all datasets used, however, the best-performing models all contained an LSTM element. Their main issues were training time and cost and the amount of data available, highlighting the importance of why we need data simulation. Training deep learning

[1]The TTS-CGAN code can be found here (https://github.com/imics-lab/tts-cgan)

models is always a significant undertaking and given our limited resources, this must be considered.

We have not been able to find a lot of previous studies of event prediction for time series specifically using autoencoders. In one recent study [Theunissen et al. 2021], they successfully applied both an autoencoder and a convolutional autoencoder to predict furnace blowback events from multivariate time series data. They compared the two deep learning models to a dynamic principal component analysis (PCA) model, which they found to be inferior. Furthermore, they concluded that the one-dimensional convolutional autoencoder outperforms the standard autoencoder. These results show us that it does seem possible to build an autoencoder, which can be used for event prediction for time series data.

Self-supervised Learning
Self-supervised learning has been shown to create a better feature representation and help combat data scarcity within several domains. In medical image classification [Azizi et al. 2021] have found that the use of multiple images and the improvement of data augmentation can further increase self-supervised pre-training and significantly outperform supervised pre-training. Additionally, pre-training such a model is more scaleable due to annotations not being a requirement.

Furthermore, this approach has also been used in emotion recognition through physiological data and is demonstrated by [Montero Quispe et al. 2022]. They have shown that self-supervised learning, through the pretext tasks in pre-training, was able to obtain a high-level representation that was successfully transferred to the downstream task of emotion recognition. The pretext tasks consist of scaling, adding noise, flipping, etc. The self-supervised model provided a result that is comparable with a fully supervised counterpart.

3.5 Summary

[Lønfeldt et al. 2023] showed the feasibility of OCD prediction, however, the results may be further improved. One suggestion from Lønfeldt et al. is transfer learning, which we will explore through two methods found in the literature, namely data simulation and the usage of deep learning models. In other domains multiple studies [Tremblay et al. 2018; F. Wang et al. 2019; Minor et al. 2020; Alkhalifah, H. Wang, and Ovcharenko 2022] found that the addition of simulated data significantly increased performance. Due to limited resources, we decided to use physiological simulation with NeuroKit2 [Makowski et al. 2021] and JOS-3 [Takahashi et al. 2021], acknowledging the limitation of the correlation between signals. Furthermore, deep learning approaches from previous studies, such as [Deznabi and Fiterau 2023] have inspired our model architecture of combining models for different frequencies to be used on multivariate physiological time series data. The addition of complex architectures also significantly increased performance in their respective studies.

4 Data

The main objective of this thesis is to explore the possibilities of predicting OCD events based on physiological data by utilizing deep learning. To do so, we have been given access to the Wrist Angel data [Olesen, Lønfeldt, et al. 2023], which is collected using the Empatica E4 wristband [Empatica 2023c]. This chapter will cover a description of the Empatica E4 wristband, as well as four types of physiological data signals and accelerometer data measured by the wristband. Furthermore, we will introduce the Wrist Angel dataset and other open source datasets that were used, which are ADARP [Sah et al. 2022], AffectiveROAD [Haouij et al. 2018], DTU-Experiment, WESAD [Schmidt et al. 2018], and WEEE [Gashi et al. 2022]. Finally, we will provide details of our simulated dataset.

4.1 Empatica E4

To ensure the data is ecologically valid, we need a method to capture data while participants are unconstrained by traditional laboratory settings. The benefit of collecting data in the wild is that it offers the opportunity to capture data from individuals in their natural environment and provides a more true-to-life basis. Therefore, wearable sensors such as wristbands and smartwatches present great opportunities for the non-invasive and convenient acquisition of multimodal signals. There exists a vast range of different brands and models of portable devices to collect signals with varying precision. Empatica's E4 wristband [Empatica 2023c] [Garbarino et al. 2014] is classified as an IIa medical device[Group 2021] in the European Union and approved by The US Food and Drug Administration (FDA)[*Empatica Legal & Compliance* 2024]. This supports the E4 wristband as a non-invasive medical device to be able to collect data in naturalistic settings. For this thesis, we will only examine the data collected from an E4 wristband. The wristband has four different sensors, as shown in fig. 4.1:

1. A photoplethysmography (PPG) sensor to measure blood volume pulse (BVP), from where heart rate (HR) among others can be derived.
2. An electrodermal activity sensor to measure electrodermal activity (EDA).
3. An infrared thermopile sensor to measure skin temperature (TEMP).
4. A 3-axis accelerometer sensor, to capture movements of the wristband (ACC).

Three signals are measured with the PPG sensor: BVP, HR, and inter-beat interval (IBI). Both HR and IBI are calculated from the BVP signal. Heart rate (HR) is extracted by peak detection on the BVP signal and then smoothing the signal with a window of 10 seconds according to the website of Empatica[1]. IBI is also derived based on BVP; however, with movement above 30% of the time, the variability of heart rate cannot be reliably calculated from the IBI signal. Due to the design and uncontrolled aspect of in-the-wild recording, IBI and heart rate variability will not be used in this thesis.

In addition to the four sensors, the wristband includes an event mark button to tag events based on the need and an internal real-time clock that timestamps the data, including a button press.

[1] https://support.empatica.com/hc/en-us/articles/360029469772-E4-data-HR-csv-explanation

Figure 4.1: Specifications of the Empatica E4 wristband [*E4 wristband technical specifications* 2024] ©Empatica

4.2 Physiological Data Description

In [Giannakakis et al. 2022] they have shown that BVP, EDA, HR, and TEMP all have significant changes during stress. Due to the limitations of the E4 wristband, these are the only four physiological signals that we can collect and use alongside the accelerometer data. In this section, we have described each of the four physiological signals, how they are detected by the Empatica E4, and their relation to stress conditions according to current literature.

4.2.1 Blood Volume Pulse (BVP)

BVP is the change in the volume of blood over time and can be detected with a PPG sensor [Empatica 2023b] and with E4 it is captured at a sample rate of 64Hz. Blood pressure can be described by different measures such as systolic blood pressure (SBP) and diastolic blood pressure (DBP), among others. In [Giannakakis et al. 2022] 15 different studies have been examined and all of them have shown a significant increase in SBP and DBP during stress.

4.2.2 Heart Rate (HR)

HR is the average number of beats in some time interval. The E4 wristband provides this signal at a sampling rate of 1Hz, which is frequently referred to as beats per minute (BPM) [P. Rautaharju and F. Rautaharju 2007]. According to [Giannakakis et al. 2022] and their review, HR is the most notable feature that significantly increases during stress. Here 23 different studies are examined, where 18 report a significant increase during stress, and the last 5 report no significant difference in HR during stress.

4.2.3 Electrodermal Activity (EDA)

EDA is a measure of variation in the electrical properties of the skin. It is a general measure, that can be split into different features, the most commonly used is the tonic (electrodermal level) and phasic (electrodermal response) part [Boucsein 2012]. Multiple features of EDA have been reviewed in [Giannakakis et al. 2022], where the studies found a significant and a nonsignificant increase during stress. Most studies (9) have examined the phasic part of EDA, which are peaks in the signal typically as a response to a stimulus. Of these 9 studies, 7 reported a significant increase during stress, and 2 reported no sig-

nificant difference. With the E4 wristband, the coupling of the electrodes with the skin can take around 15 minutes and is measured at a sampling rate of 4 Hz.

4.2.4 Skin Temperature (TEMP)

TEMP is the temperature of the skin. They are reported in degrees Celsius (°C). From the E4 wristband, TEMP is measured on either the left or right wrist with a sampling rate of 4Hz. [Giannakakis et al. 2022] reviewed studies showing either a significant increase or decrease in skin temperature during stress conditions. These skin temperatures are reported for various locations on the body, but none of them specifically around the wrist. One study specifically looking at wrist TEMP and stress did not find significant differences in TEMP during stress [Vinkers et al. 2013].

4.2.5 3-axis accelerometer (ACC)

The E4 wristband has a 3-axis accelerometer sensor in the range $[-2g, 2g]$. The data captured with the accelerometer provides the x, y, z directions of the acceleration and has a sampling frequency of 32Hz. According to E4's official website [Empatica 2023a] the raw data can be interpreted for analytical purposes by $xg = x * 2/128$, meaning an x of 64 is equal to $1g$. The accelerometer data will not be used for OCD prediction but for a separate physical activity classification.

4.3 Data from the Wrist Angel study (WA)

The data from Wrist Angel [Olesen, Lønfeldt, et al. 2023], comes from an in-the-wild experiment. 18 youths aged 8-16 and their parents participated in the experiment. Nine of the youth participants were diagnosed with OCD and the remaining nine participants were a control group without a psychiatric diagnosis. In this thesis, we only look at the youth of the experimental group, consisting of children and adolescents with OCD. The participants wore the Empatica E4 wristband on their non-dominant hands during waking hours for eight weeks. Thus, producing data on the four signals from the E4 wristband (BVP, HR, EDA, and TEMP). Furthermore, the participants were asked to tag OCD events by pressing the "Event Mark Button" fig. 4.1 on the wristband each time they felt stressed by OCD symptoms. Besides the four physiological signals we retrieved from the E4 wristband, we also had access to accelerometer data in three dimensions (x,y,z). An example of data available from the Wrist Angel dataset can be seen in fig. 4.2.

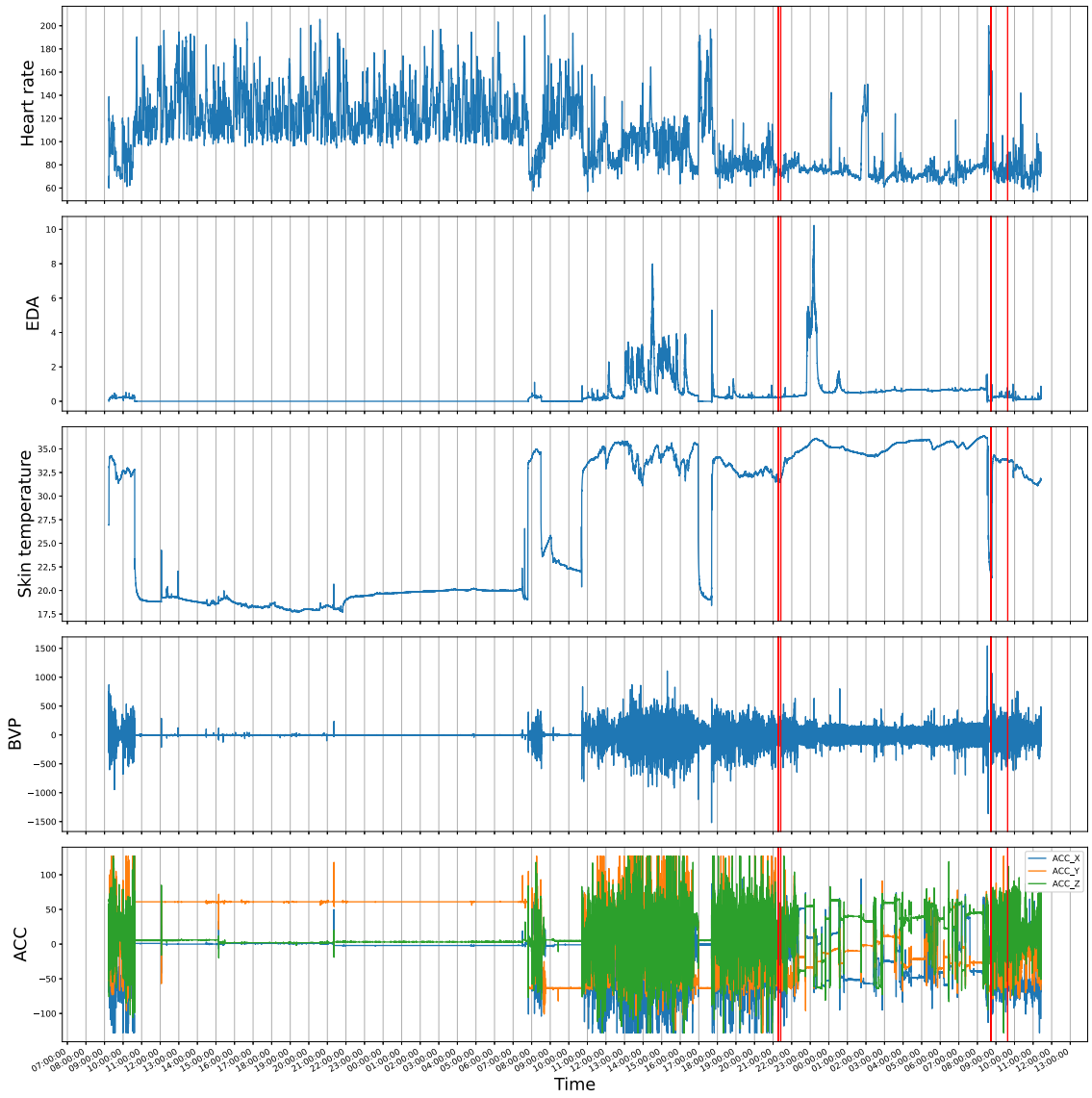
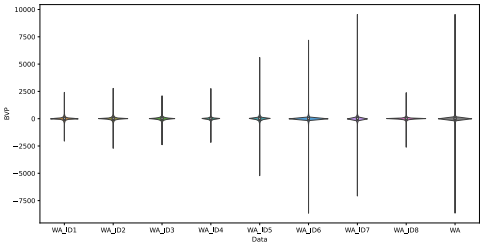
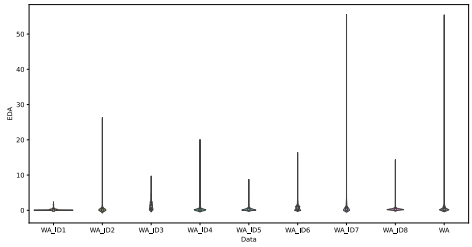


Figure 4.2: This figure is taken from the supplementary material Figure S.2 from [Olesen, Das, et al. 2023]. It is an example of a recording from the Wrist Angel dataset. The red lines indicate tagged OCD events.

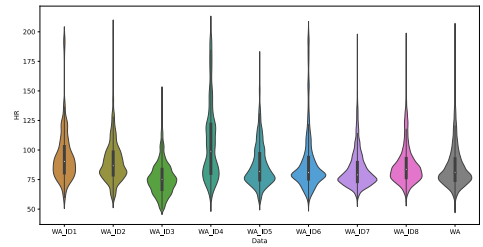
Since the data comes from different participants we have plotted violin plots for each signal based on the participant in fig. 4.3.



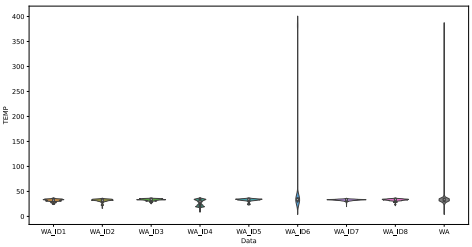
(a) BVP on the y-axis of each ID.



(b) EDA on the y-axis of each ID.



(c) HR on the y-axis of each ID.



(d) TEMP on the y-axis of each ID. A similar plot where the 102 outliers are removed can be seen in appendix A.7

Figure 4.3: Violin plots of the four physiological signals for each participant. The last violin plot of each signal is a combined plot of the entire Wrist Angel dataset.

From fig. 4.3 it can be seen that there are some differences in physiological signals between participants, but overall they seem to have the same tendencies, hence it should be possible to generalize. However, we do see some outliers for skin temperature. These outliers are 102 observations from ID 6, where the TEMP is fixed at 380.65°C. They all come from the period from day 21 to day 27. We looked at the signals from these outliers[2]. In these observations, the HR is also high, between 120 and 150. BVP and EDA look like resting, without significant peaks. The acceleration varied, some with movements, and some without movements. Overall it did not seem like all signals were off, but perhaps the temperature sensor was broken or not calibrated. These observations have not been removed from the dataset because we use the same preprocessing as in [Lønfeldt et al. 2023], see section 5.2.

Additionally, we had access to laboratory recordings. In these, the control group (with parents) was part of a small experiment, where they wore the wristbands while resting and performing physical activity, respectively. Therefore, we have physiological data (which we did not use) and accelerometer data from physical activity, which was used for the classification of physical activity.

[2]The plots are not included in this thesis due to privacy concerns.

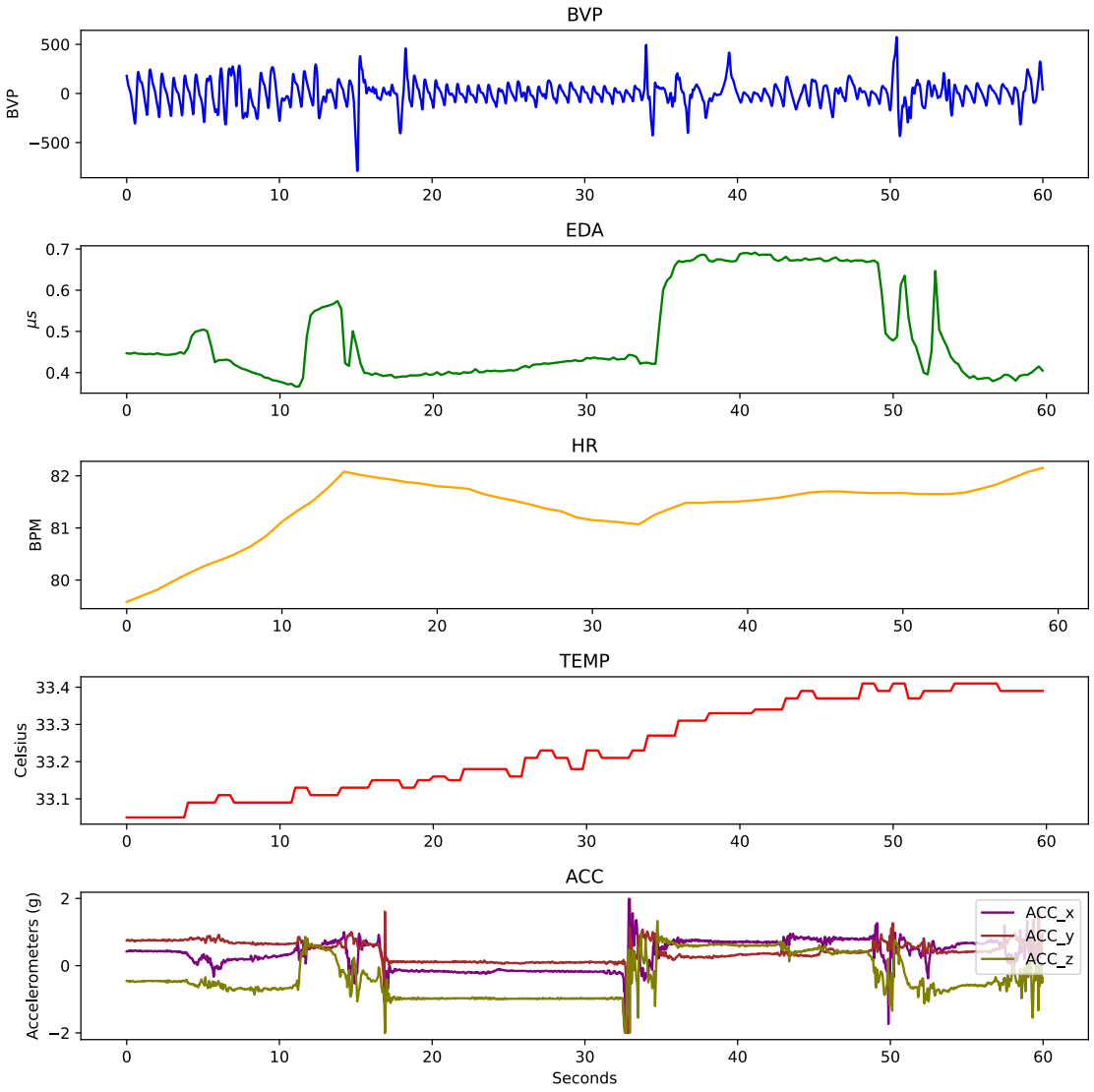
4.4 Open-Source Dataset

Besides the Wrist Angel data, we used five different, open-source datasets, which are described in the following sections. The first four datasets ADARP, ROAD, DTU, and WESAD (sections 4.4.1 to 4.4.4) are stress datasets used to pre-train our OCD prediction model. The last dataset, WEEE in section 4.4.5 is an activity dataset used for the classification of physical activity.

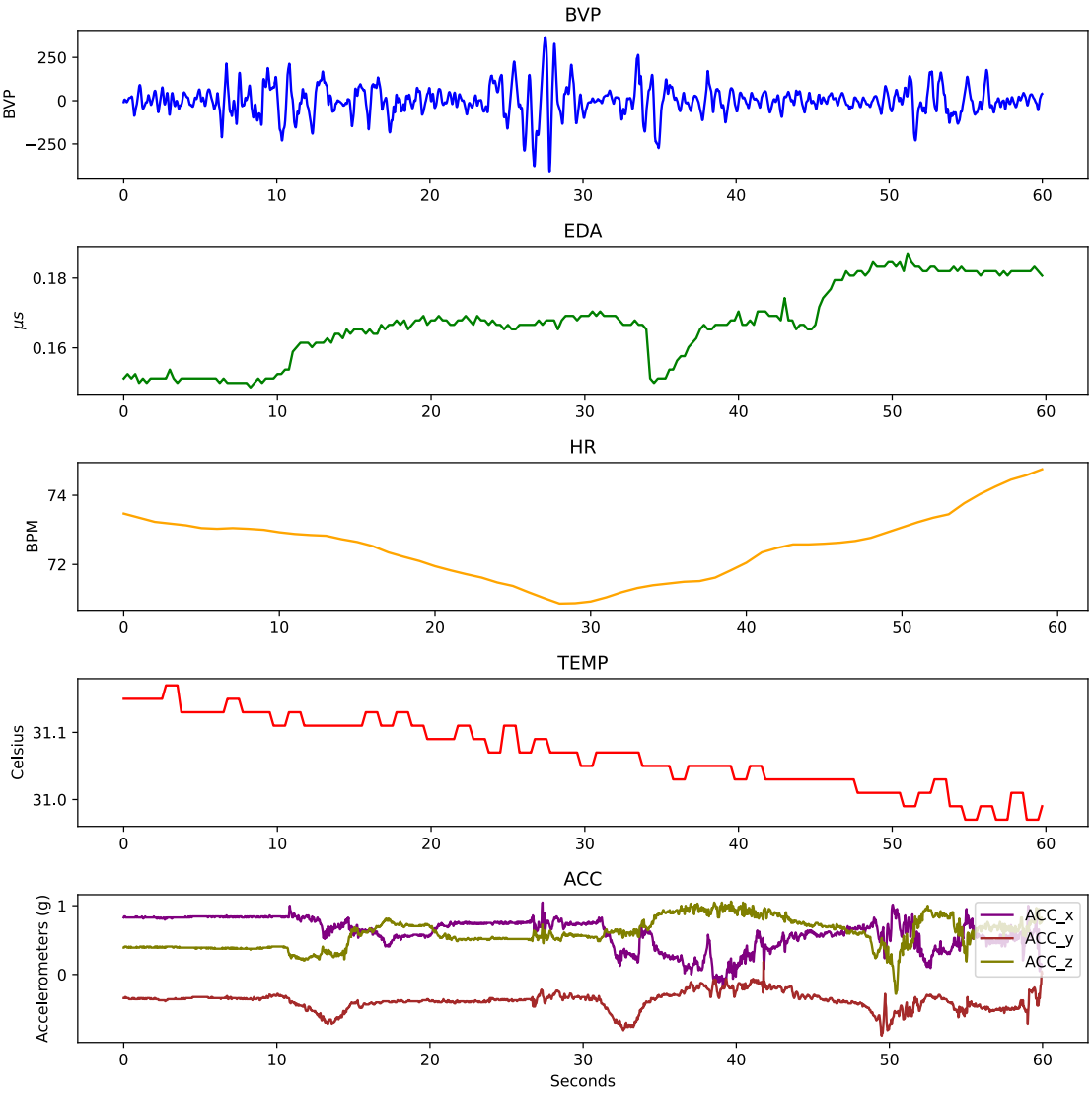
4.4.1 ADARP: A Multi Modal Dataset for Stress and Alcohol Relapse Quantification in Real Life Setting

In [Sah et al. 2022] a proof-of-concept study was conducted at Washington State University, to examine, whether a wearable sensor can be used to detect stress in patients suffering from alcohol use disorder (AUD). 11 participants (10 women) were recruited and all received mental health and AUD treatment at a treatment agency in the state of Washington. Participants were on average involved in the study for 14 days. During the initial session, the participants were given an E4 wristband and told to wear it during the day and only take it off when sleeping or at times when the device could be damaged. The participants were instructed to press the event button when they felt “more stressed, overwhelmed, or anxious than usual” [Sah et al. 2022]. Furthermore, there was an ecological momentary assessment (EMA) questionnaire sent out four times a day, however, this data will not be used in our study. In total 1698 hours of physiological data were collected using E4, with an average of 11.5 hours each day and the participants tagged 409 events in total. There is a limitation of this dataset, participants in this study are diagnosed with AUD, which will result in physiological signals different from those of an average person. In [Alinia et al. 2021], they further evaluated the data and found that physiological signals were significantly associated with several self-reported outcomes from the participants.

An example of an extracted stress and non-stress observation from the ADARP dataset can be seen in fig. 4.4.



(a) A stress event observation.



(b) A non-event observation.

Figure 4.4: An example of an event and non-event segment with a duration of 1 minute for participant 8 in the ADARP dataset. Each signal is from the Empatica E4 wristband, BVP, EDA, HR, TEMP, and acceleration. Additional visualizations of other datasets can be seen in appendices A.1 to A.3

4.4.2 AffectiveROAD Dataset (ROAD)

The AffectiveROAD dataset [Haouij et al. 2018] is from an experiment with 14 drives, from 10 different participants. All participants drove the same route on three different types of roads. Each participant wore two Empatica E4 wristbands, one on each wrist, together with various other sensors, which are not relevant in this study. We limit our use to the E4 wristband on the left wrist since for most experiments the participants were asked to wear the wristband on their nondominant hand. From the wristband, we obtain the four signals (BVP, HR, EDA, TEMP) from Empatica E4 over the full drive, where the driver has been exposed to different stressful driving situations. Additionally, the drivers were observed and subjectively evaluated on a stress metric between 0 and 1, where 0 was not stressed, and 1 was extremely stressed. These scores were then validated by the driver after the session. An example of the ”stress” metric can be seen in fig. 4.5.

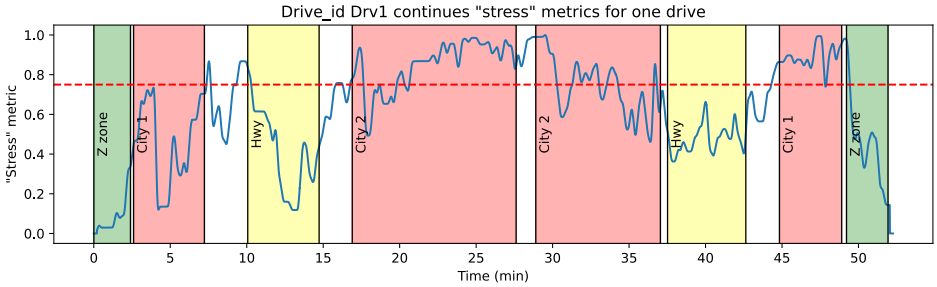


Figure 4.5: The subjective ”stress” metric from zero to one over time in minutes. Provided by the observer and validated by the driver. The color shows the different types of roads annotated from the data. The horizontal dashed line at 0.75 is the threshold for classifying the data as stress. The figure is inspired by [Bustos et al. 2021].

In [Bustos et al. 2021], they min-max normalized the data to the range 0 and 1 for the stress metric and divided it into three categories, low stress below 0.4, medium stress between 0.4 and 0.75, and high stress above 0.75. We took inspiration from their division but only needed two classes; hence we used 0.75 as our threshold for stress, resulting in only classifying high stress periods.

4.4.3 DTU-Experiment

The DTU-Experiment dataset [3] is collected through an experiment conducted at the Technical University of Denmark (DTU). A total of 28 participants spread over three cohorts participated in the experiment. A visualization of the experiment design can be seen in fig. 4.6. There were four rounds of the experiment. For each round, the participant went through three 5-minute phases. The first phase was a pre-task resting phase, the second phase was task-solving, and the final phase was a post-task resting phase. The puzzle they had to solve was a tangram task, which was originally used to study social behavior between parents and children [Hudson and Rapee 2001]. They did it in teams of two, where one participant acted as an instructor and could see the solution but was not allowed to touch and assemble the puzzle, and the other participant acted as the puzzle-solver and could not see the solution. We categorize the signals from the task-solving phase as stress and the two resting phases as non-stress. During the experiment, all participants wore an Empatica E4 wristband. Hence from each participant, we have our data comprised of BVP, HR, EDA, and TEMP for each round and phase.

[3] https://github.com/DTUComputeStatisticsAndDataAnalysis/Analysis-of-emotions-using-physiological-signals-a-pilot-study/blob/main/Analysis_plan___biosensor___revised.pdf

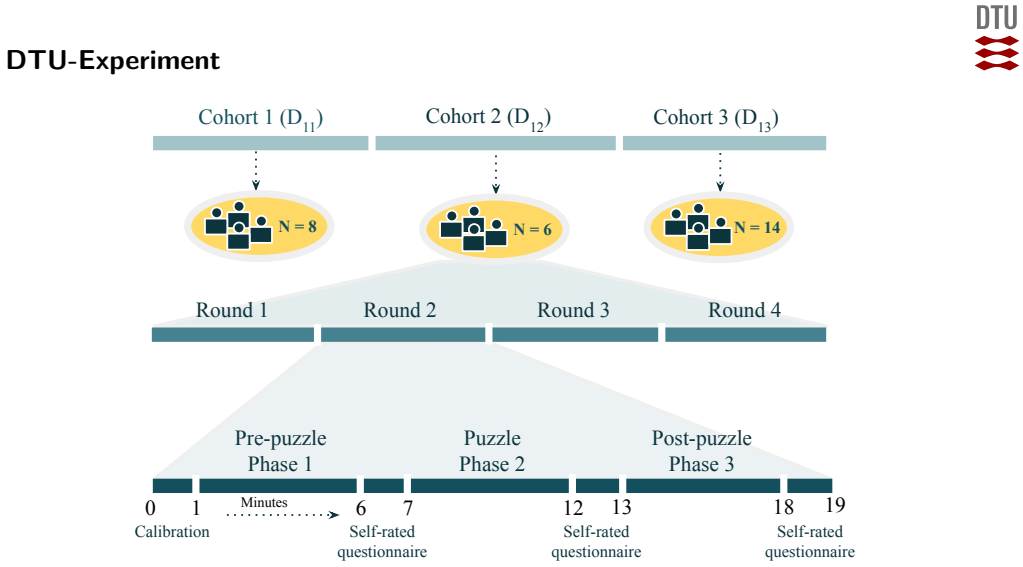


Figure 4.6: The figure is taken from the course material from DTU's course 02582 Computational Data Analysis[4]. It shows how the DTU experiment is designed.

4.4.4 WESAD: Multimodal Dataset for Wearable Stress and Affect Detection

WESAD [Schmidt et al. 2018] is another dataset that combines physiological data with stress exposure. This dataset contains data from 15 participants. Each participant wore, among other sensors, an Empatica E4 on their non-dominant hand. The data was collected over two hours, in which the participants went through different scenarios. In the stressful scenario, they were exposed to the Trier Social Stress Test (TSST) [Kirschbaum, Pirke, and Hellhammer 1993]. Here, they had to do a 5-minute speech, which the participants were told could boost their career options. Thereafter, they needed to count down from 2023 in steps of 17 and start over if they made a mistake. Data from this situation are classified as stress. Furthermore, there was a baseline scenario, in which participants were sitting/standing at a table and provided with magazines containing neutral content. Additionally, they had to watch a set of funny video clips in the amusing scenarios. The baseline scenario and the amusement scenario are combined into a non-stress class. In [Schmidt et al. 2018] they, among other things, performed a binary classification task. With their classification, they obtained an accuracy of around 0.88 and an F1 score of 0.86 using physiological data from the wrist in a random forest model.

4.4.5 WEEE: A Multidevice and Multimodal Dataset for Human Energy Expenditure Estimation Using Wearable Devices

WEEE [Gashi et al. 2022] is a dataset with physiological and accelerometer data from 17 participants. Each participant wore several different measuring devices, amongst them the Empatica E4, during three different activities: Resting, cycling on an exercise bike, and running on a treadmill, each for approximately 10 minutes. Resting activity involved sitting and standing. Cycling and running activities were carried out at two different intensity levels. In this thesis, we use only accelerometer data from the WEEE dataset. An example of accelerometer data from one participant divided into different activities can be seen in fig. 4.7.

[4] https://kurser.dtu.dk/course/02582

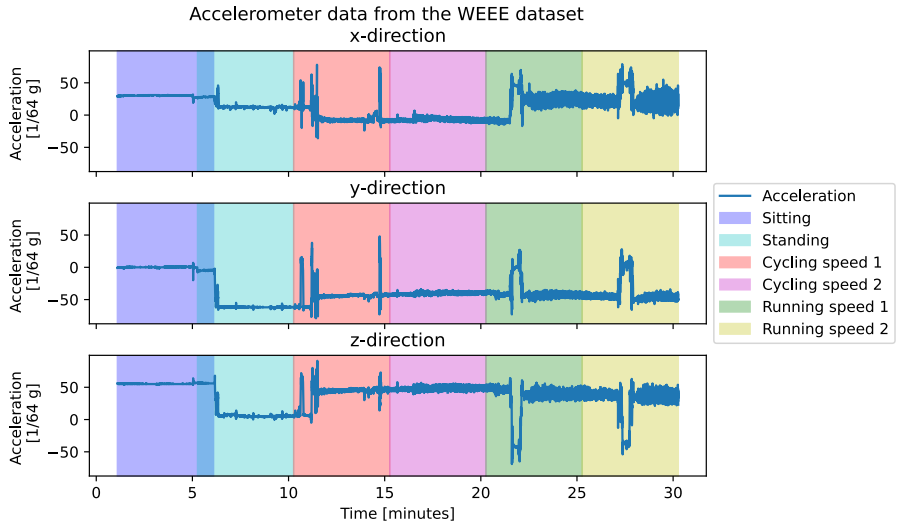
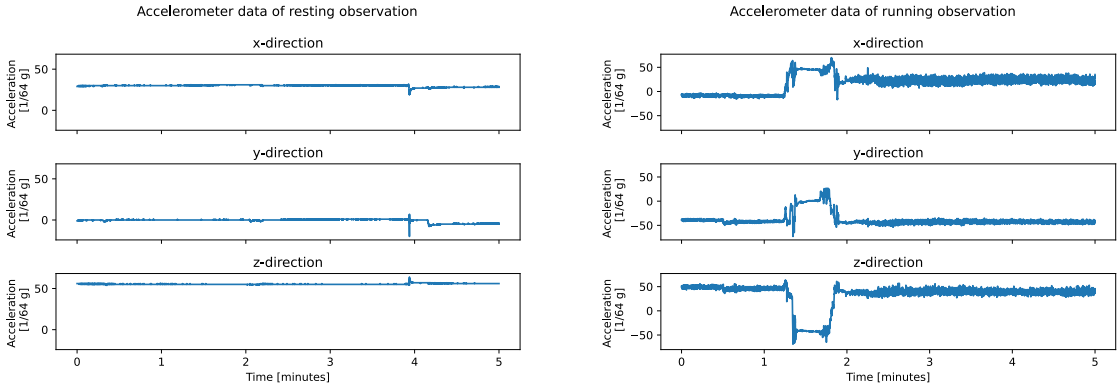


Figure 4.7: Raw accelerometer data from the WEEE dataset of one participant. The plot is divided into six different sections representing different activities.

This accelerometer data is used to find the correct threshold for activity classification. As can be seen in fig. 4.7, there is a difference in acceleration between resting (sitting and standing) and running activities. For cycling activities, smaller variations are seen in the acceleration compared to the running phases. Therefore, we only used the running activity phase as a physical activity class for classification. In practice, we tried to create as many 5-minute windows as possible for the resting phase (sitting and standing) and activity phase (running speed 1 and running speed 2) with no overlap. The dataset contained 16 of the 17 participants. One participant (ID 14) was not included, because the starting times of the activities cycling and running are similar. For each of the 15 participants, we were able to obtain two rest observations and two activity observations. For the participant with ID 3, we were only able to collect two rest observations and one activity observation, due to limited data from this participant during the physical activity phase. In total, we were able to obtain $16 * 4 - 1 = 63$ observations, where 32 are resting observations and 31 are physical activity observations. Examples of a resting observation and a physical activity observation can be seen in fig. 4.8.



(a) Resting observation.

(b) Physical activity observation.

Figure 4.8: Accelerometer data for two random observations, respectively resting and physical activity.

4.5 Comparison of Datasets

To compare the five different OCD/stress datasets used throughout this report, histograms of the mean and standard deviation have been plotted in figs. 4.9 and 4.10. Similar histograms can be seen for signal-to-noise ratio in appendix A.4.

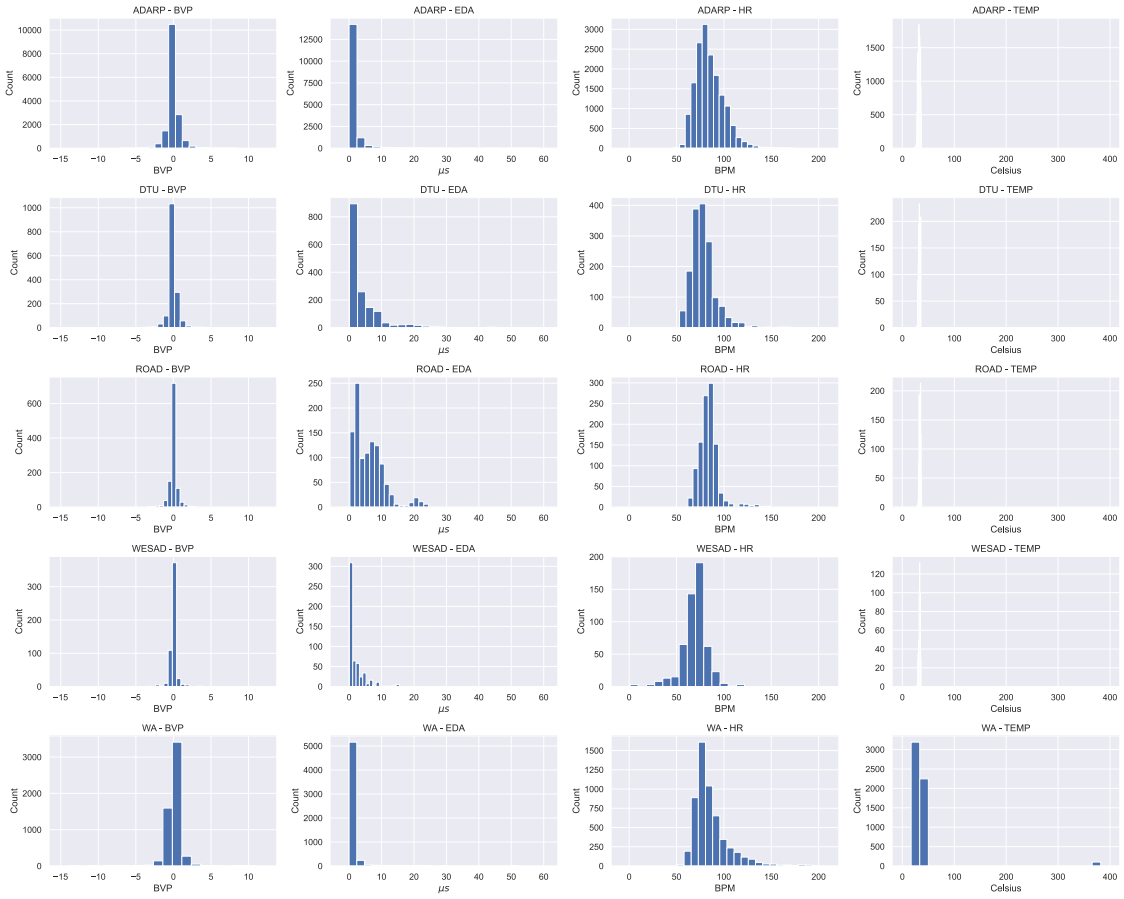


Figure 4.9: Histograms of the observations' mean for each signal and each dataset. The columns are the signals, from left to right: BVP, EDA, HR, and TEMP. The rows are the datasets, from top to bottom: ADARP, DTU, ROAD, WESAD, and WA. For better visibility of the histogram, a similar plot where the outliers have been removed can be seen in appendix A.5.

From the histograms of the means in fig. 4.9, all datasets seem quite similar, even though the ROAD dataset has some higher values for EDA than the other datasets.

From the standard deviation histogram in fig. 4.10, the datasets look mostly similar as well. The main difference is the standard deviation of HR for the WESAD dataset seems more varied and has higher standard deviation values than for the other datasets. This can also be seen in the example observations plotted in fig. 4.4 and appendices A.1 to A.3. Additionally, WESAD has a more spiky HR signal compared to the other datasets. Since WESAD was the first dataset collected, a version update of the E4 software could be responsible for the difference in this signal.

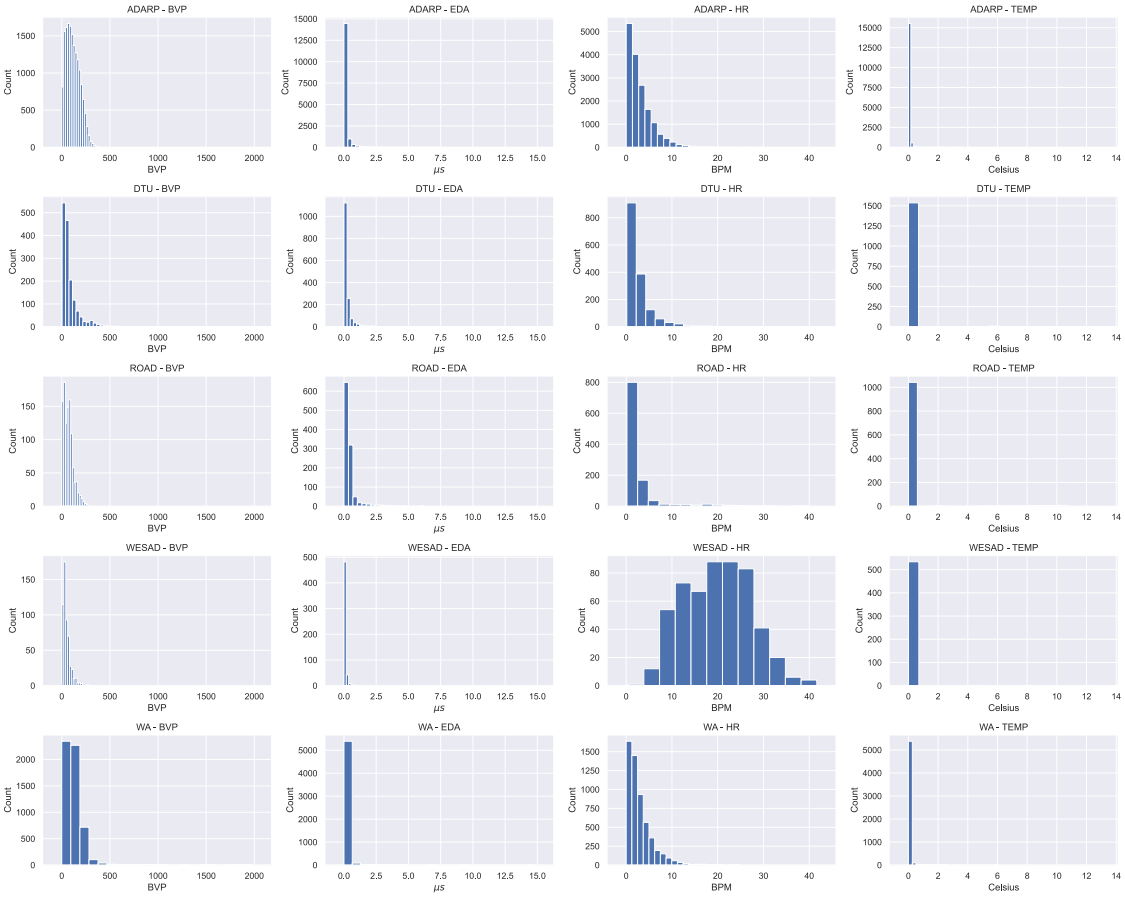
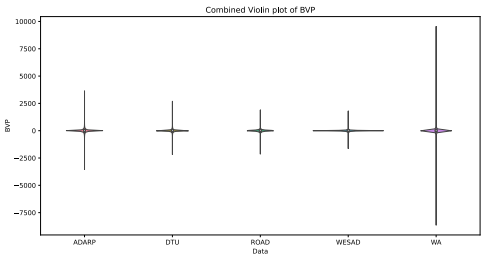
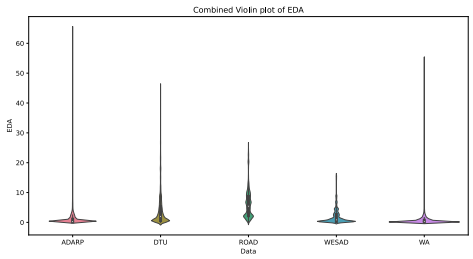


Figure 4.10: Histograms of the observations' standard deviation for each signal and each dataset. The columns are the signals, from left to right: BVP, EDA, HR, and TEMP. The rows are the datasets, from top to bottom: ADARP, DTU, ROAD, WESAD, and WA.

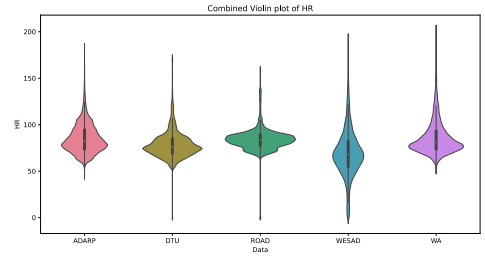
Finally, for further comparison, we have also plotted violin plots for the different signals between datasets as seen in fig. 4.11.



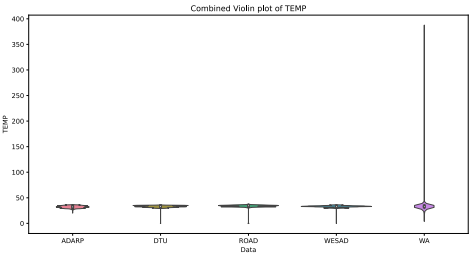
(a) BVP violin plot for each signal.



(b) EDA violin plot for each signal.



(c) HR violin plot for each signal.



(d) TEMP violin plot for each signal. A similar plot where the 102 outliers are removed can be seen in appendix A.6

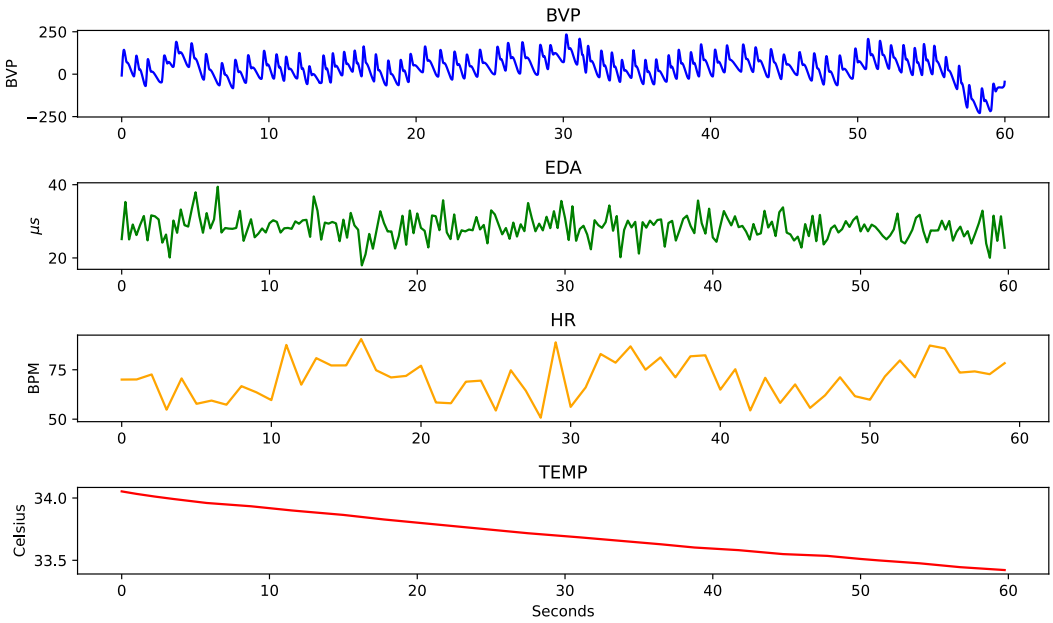
Figure 4.11: Violin plots of the four physiological signals for each dataset. The datasets are from left to right: ADARP, DTU, ROAD, WESAD, and WA.

From the violin plot of BVP fig. 4.11a it can be seen that the Wrist Angel dataset has a higher range than the other datasets, and especially WESAD has a low range and most observations around zero. For EDA fig. 4.11b the same trend can be seen, as these in-the-wild datasets (Wrist Angel and ADARP) have higher ranges than the other datasets. For the HR we see different distributions, but no distinct differences. The same goes for TEMP; a plot with outliers removed can be seen in appendix A.6.

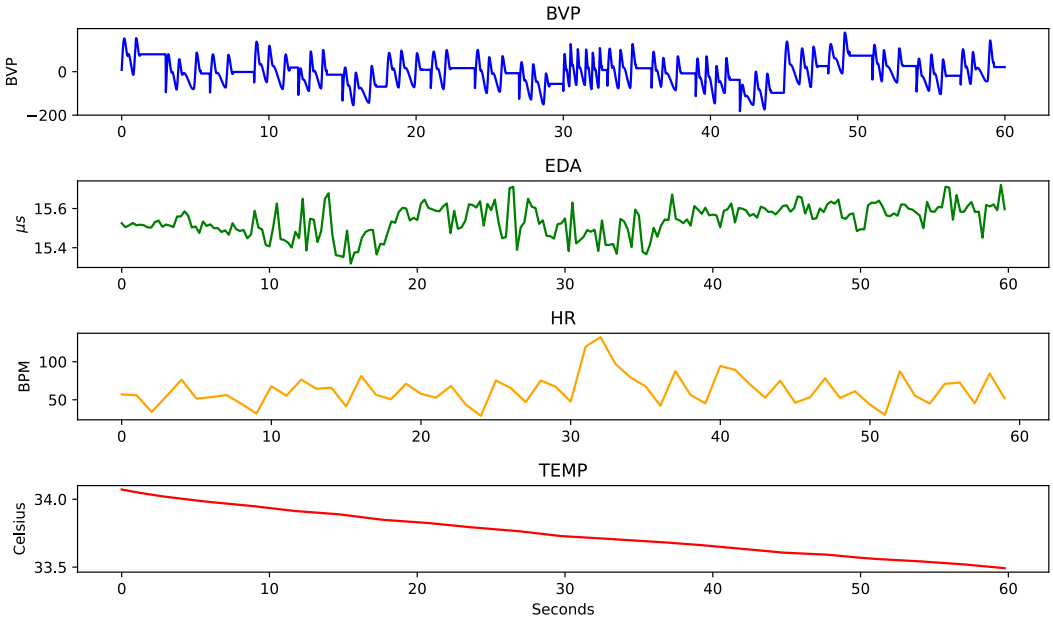
Generally, when comparing the five datasets, we do see some differences between them. However, by the comparison methods we used, there are no large differences that indicate that the open-source datasets cannot be representative of the Wrist Angel dataset. Additionally, we have also looked at various examples of the signals, and as non-medical personnel, we cannot distinguish between the different datasets. Hence based on these findings, we see no roadblocks for transfer learning to work.

4.6 Simulated Data

We created two different simulated datasets to test if training a deep learning model on simulated data would improve performance. The two datasets were created with different methods, the first is a plain method and the second a fragmented method, which will both be explained in depth in sections 5.1.1 and 5.1.2.



(a) The four simulated signals using the plain method.



(b) The four simulated signals using the fragmented method.

Figure 4.12: The 60 seconds of data is simulated for a 30-year-old male with a normal physical condition in spring. For each simulation we have randomly selected variables such as average heart rate and scale, thus for the two different methods, these variables differ. A version with a duration of 5 minutes can be seen in Appendix fig. A.8.

4.7 Summary

Throughout this project, we only worked with data from the Empatica E4 wristband and simulated data. Therefore, all data had a similar structure, and the four physiological signals we used to predict OCD events were blood volume pulse (BVP), heart rate (HR), electrodermal activity (EDA), and skin temperature (TEMP). Furthermore, acceleration, which was also captured by the E4 wristband, is used for physical activity classification. The primary dataset used to evaluate our model is the Wrist Angel dataset containing in-the-wild data from children and adolescents with OCD. In addition to this, four open-source datasets, ADARP, ROAD, DTU, and WESAD, were used for initial testing and to pre-train our model. Between these five OCD/stress datasets, there seem to be no large differences, which should prevent the usage of the datasets together. An additional dataset, the WEEE dataset, was used to create a model for the classification of physical activity. Finally, we created two simulated datasets, which we tested for pre-training, but did not use with the final pre-trained model.

5 Methods

In this chapter, we present different methods used throughout this thesis. First, we will describe the two methods we created to simulate data: plain and fragmented. Then we present our preprocessing pipeline and how it handles the datasets differently. Additionally, up- and downsampling techniques, alongside filtering and standardization, are also presented. Then follows our evaluation and statistical test methods. Hereafter all different model architectures were used along with our final model being showcased. Subsequently, we explain how we aim to utilize transfer learning through fine-tuning a pre-trained model and self-supervised learning. Two activity classification models used for filtering are also introduced before finally we describe and illustrate how all the different factors explained throughout this method section are applied to the Wrist Angel dataset.

5.1 Simulating Data

To increase the performance of deep learning models, we proposed training the models on both open-source data and simulated data. The purpose of the simulated data was to mimic the data retrieved from an E4 wristband since all the data used came from an E4 device. Furthermore, we chose to focus only on 4 different types of physiological signals, namely: BVP, HR, EDA, and TEMP.

To ensure the representativeness of the signals, we simulated the data uniformly across 66 different ages and various conditions, as shown in table 5.1. BVP, EDA, and HR are simulated with Neurokit2, and the ages are represented based on the parameters specified in table 5.2. TEMP was simulated with JOS-3, where age was an input parameter. We chose the minimum simulation age to be 5, as this is the age when children can start to wear watches and similar devices [GMTLondon 2023]. We used every rounded age until the age of 70, and this limit was mainly set to spare resources. In reality, physiological signals have been recorded for those over 70 years of age, but most experiments are conducted with younger participants. Nevertheless, this would not have a big effect on our results as our model would not be used for elderly people. Furthermore, we simulated data from five different weather conditions, as this can affect TEMP. These conditions were based on the four seasons, and the fifth condition was indoors, assuming a similar environment indoors throughout the year. The three different physical conditions were just a simple grouping of people of different physical builds because this can affect some of the physiological signals such as HR [Hipp 2023]. Finally, all simulations were performed for both men and women. This gave us a total of $66 \times 5 \times 3 \times 2 = 1950$ different settings to simulate.

Variable	Description	Number of classes
Age	All ages from 5 to 70 years old	66
Weather	Winter, Spring, Summer, Autumn, and indoors	5
Physical condition	Fit, normal, or poor shape	3
Gender	Female or male	2

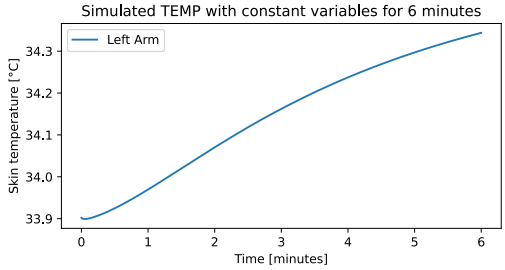
Table 5.1: The variables we use to create a grid for simulating data for each unique combination of variables. In total, we have $66 \cdot 5 \cdot 3 \cdot 2 = 1980$ different combinations of conditions to simulate. These variables are realized based on the parameters presented in table 5.2.

	Parameter	Range/Values	Step size
BVP & HR			
	Standard deviation HR noise	[-1, 1]	0.1
	Physical condition effect on average HR	{good = -5 normal = 0, poor = 5}	
	Physical condition effect on standard deviation HR	{good = 10, normal = 8, poor = 6}	
	Genders effect on HR	{female = +8, male = 0}	
	The average HR for 5 to 9 years old	[70, 115]	1
	The average HR for above 10 years old	[60, 100]	1
	Scale value for BVP	[0.001, 500]	0.001
EDA			
	Drift for EDA	[-0.1, 0.1]	0.001
	SCR peak	True/False with a probability of 0.1 for true	
	EDA amplitude of laplace noise	[0.2, 2]	0.01
TEMP			
	BMI for good physical condition	[16, 27]	0.01
	BMI for normal physical condition	[18.5, 25]	0.01
	BMI for poor physical condition	[15.5, 18.5] $\cup$ [25, 40]	0.01
	Weight	BMI * height* height	
	Height for 5 to 9-year-olds in m.	[1.03, 1.39]	0.01
	Height for 10 to 14-year-olds in m.	[1.29, 1.7]	0.01
	Height for female above 15 year olds in m.	[1.5, 1.9]	0.01
	Height for male above 15 year olds in m.	[1.65, 2.05]	0.01
	Humidity based on weather	{winter = 89, spring = 85, summer = 78, fall = 76, indoor = 45}	
	Temperature for winter	[-18, 12.5]	0.5
	Temperature for spring	[-15, 25]	0.5
	Temperature for summer	[1.5, 32]	0.5
	Temperature for fall	[-6, 24]	0.5
	Temperature for indoor	[15, 28]	0.5
	I_{cl} for winter	[1.10, 1.5]	0.01
	I_{cl} for spring	[0.5, 1.1]	0.01
	I_{cl} for summer	[0.3, 0.8]	0.01
	I_{cl} for fall	[0.5, 1.1]	0.01
	I_{cl} for indoor	[0.35, 0.9]	0.01
	CI based on physical condition	{good = 4.2, normal = 3.5, poor = 2.8}	
	Wind speed indoor	0.1	
	Wind speed not indoor	[0, 25]	0.1
	Posture	[sitting, laying, standing]	
	Activity	[1, 4.4]	0.1

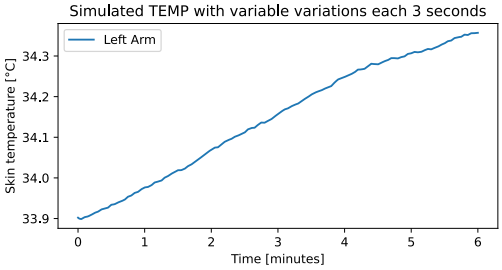
Table 5.2: The parameters for simulating data using Neurokit2 and JOS-3. The parameters are defined to realize our variables in table 5.1. The explanation for the chosen parameter ranges can be seen in appendix A.9.

For BVP, EDA, and HR we had two different ways of simulating the data. The first method will be named the plain method, which mostly uses `NeuroKit2` created by [Makowski et al. 2021]. The second method will be called the fragmented method because we simulated signals in small intervals.

For both methods skin temperature (TEMP) was simulated through the same steps, using the Python package JOS-3 (Joint system thermoregulation model) [Takahashi et al. 2021]. JOS-3 is developed based on the previous JOS-2 model [Kobayashi and Tanabe 2013], which is based on the Stolwijk model [Stolwijk et al. 1971]. Hence JOS-3 is a mathematical multi-node model of physiological body temperature regulation of the entire body. The amount of nodes is increased from 25 nodes in Stolwijk's original model to 85 nodes in JOS-3. Furthermore, the body is divided into 17 segments, among others are the right and left arms, which was relevant for us since the E4 wristband was placed there [*E4 sensors* 2023]. The JOS-3 model requires specifications of each person including their height, weight, age, sex, and cardiac index. In addition, it is possible to set specific external conditions such as temperature, mean radiant temperature, air speed, and humidity. Finally, it is also possible to specify personal conditions such as clothing insulation, physical activity, and posture. An overview of all possible inputs used in simulating data can be seen in table 5.2. The model can be defined with the `JOS3` class, and data can then be simulated with the `.simulate` function for a specified time and frequency. Furthermore, external and personal conditions can be redefined with the model to simulate a change in condition. In practice, we simulated the data by simulating small intervals and combining them. Initially, we defined our setting based on the parameters described in table 5.1. We then simulated three seconds of data and randomly changed the following conditions, temperature ($\pm 2°C$), radiant temperature ($\pm 2°C$), humidity ($\pm 5\%$), physical activity (± 0.4), simulate again, and so on.



(a) A 6-minute simulated TEMP signal using JOS-3, without varying any variables during the simulation.



(b) A 6-minute simulated TEMP signal using JOS-3, where variables have been randomly varied every three seconds.

Figure 5.1: Plot of the simulated TEMP using JOS-3, where variables are kept constant, and the described approach of simulating 3-second intervals of TEMP data and varying variables for each interval. It can be seen that varying variables add noise to the data.

As seen in fig. 5.1, by varying variables every three seconds it makes the data, a bit more noisy to imitate real data better. We use a frequency of 4Hz, which is the same as the data from E4.

5.1.1 Plain Method

As the name suggests the plain method mainly uses the default `NeuroKit2's` various simulation functions for BVP, EDA, and HR. BVP is the signal with the highest sampling rate from the E4 wristband (64Hz) which is captured by photoplethysmography (PPG) sensors. Furthermore, on E4's webpage [Empatica 2023b], the BVP signal from the wristband has some specific properties. Firstly, the value usually has a range of $[-500, 500]$, and secondly, the signal has a zero mean. Thus, this physical signal can be simulated using Neurokit2's function `ppg_simulate`. This function simulates a signal from a photoplethysmography using a phenomenological approximation of PPG. The input `heart_rate` describes the average HR for the simulated observation and will vary based on the person we are trying to simulate for, and the value is selected based on table 5.2 and appendix A.9. The BVP and HR signals are connected, thus the HR can be calculated based on the BVP value rather than simulating an unrelated HR signal. Using the interbeat interval, we can transform it into beats per minute, subsequently, we will down-sample the data to 1 Hz to mimic the data from the E4 wristband. After extracting the heart rate we scaled the BVP signal such that it has a mean of 0 and a max of a randomly selected value, as seen in table 5.2.

The final signal to simulate is EDA, this can be accomplished by using Neurokit2's `eda_simulate`. When we used this function, we started by simulating data with a sampling rate of 1000 Hz. The next step was to scale the data such that it had a maximum value above 0 and a minimum below 20. The final step was to downsample the data to 4Hz using the `scipy.signal.resample` function, which will be described in section 5.2.6.

5.1.2 Fragmented Method

To create more real-world-like PPG data, we chose to first simulate electrocardiogram (ECG) data using ECGSYN described by [McSharry et al. 2003] to obtain HR for a 9-second segment. Using these HR segments, we further divided it into 3-second intervals. The HR value in the first second of each segment was used as `heart_rate` input for the function `ppg_simulate` to simulate 3 seconds of BVP data at 64Hz. This process continued by concatenating each interval together until we had simulated the desired interval. Lastly, we scaled the BVP data in the same way as we did in the plain method. A visualization of the process can be seen in fig. 5.2.

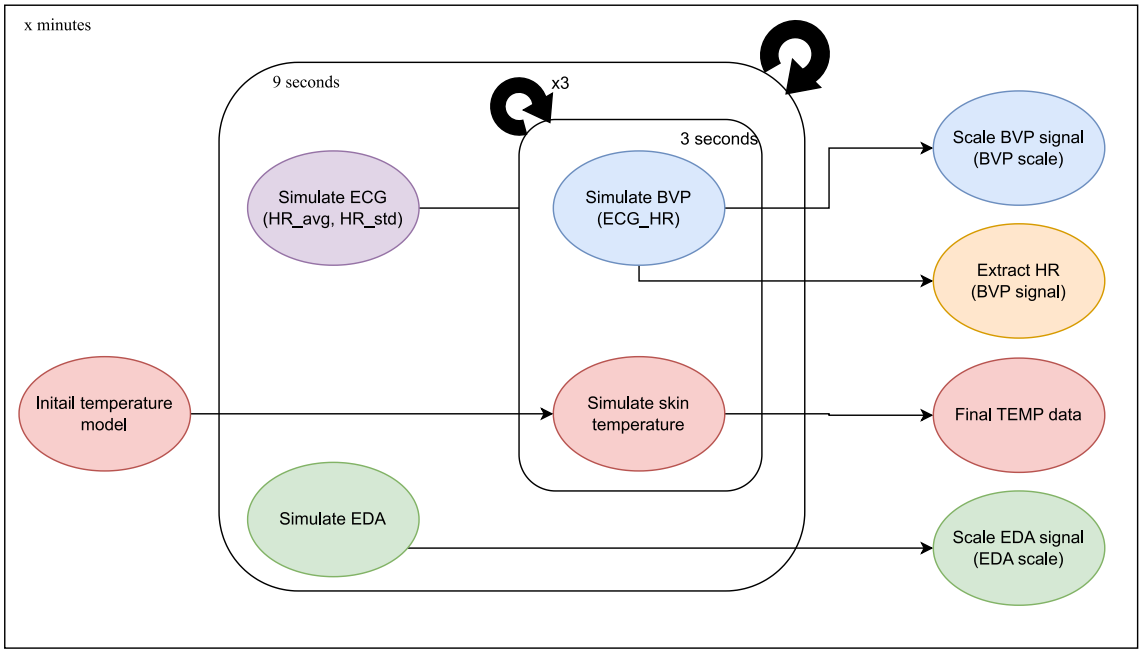


Figure 5.2: Process of simulating BVP and HR data for a given interval of *x* minutes using the fragmented method. First, we simulate ECG signals for every 9 seconds. Then HR for each of the three 3-second intervals is used as input to simulate 3 seconds of BVP signals. When all x minutes of BVP signals are simulated the HR is extracted from BVP signals based on the peaks.

The EDA signals were simulated using Neurokit2's `eda_simulate`. A similar approach was selected to increase the noise in the data such that it was more real-world-like, just as for BVP. Intervals of 9 seconds were used to simulate the data with changed input parameters. The data was simulated with a sampling frequency of 1000 Hz. To mimic the data from the E4 watch, downsampling was applied. The downsampling method we selected was to use the 1st, 333re, 666th, and last data points for each second as the EDA data, which resulted in 4 data points per second while preserving the noise. In table 5.2 it states that for EDA 3 parameters were selected to be changed: drift, SCR peak, and noise. The variable drift is the slope of the linear drift of the data, with a default value of -0.01. We chose to randomly select a drift for every 9 seconds to simulate the different linear drifts the data could have. The SCR peaks were defined to occur with a probability of 10%. The last parameter we had is the noise that describes the amplitude of the Laplace noise which was also randomly selected for each 9-second interval.

5.2 Data Preprocessing

We will introduce the different preprocessing steps that were applied to the five datasets described in sections 4.3 and 4.4. The experimental design of the five different datasets differs to various degrees. However, we can group the dataset into three categories: phase-based dataset (WESAD & DTU), in-the-wild dataset (ADARP & Wrist Angel), and stress metric dataset (ROAD). Each of these categories had their separate preprocessing steps. These steps had the purpose of obtaining an identical structure of the datasets to be used for machine learning. Furthermore, we removed data that can be considered outliers, and manual inspection was done to ensure that we did not remove more data than was necessary. Two different segment lengths were extracted for each dataset, which were one and five minutes. Throughout the thesis, we will be addressing these

datasets with different window lengths as *dataset x*-min (for example WESAD 1-min). For all phase-based and stress-metric datasets, we added zero padding if the segment was too short, but if the segment was missing more than 20%, we discarded the entire segment.

5.2.1 Phase-Based Datasets: WESAD and DTU

The experimental designs for WESAD and DTU are similar, since they both are phase-based, meaning the participants go through phases that are designed to trigger a predefined emotion.

For both stress and non-stress phases the data was designated as stress or non-stress from the start until the end of the phase. Observations were extracted from each phase and labeled accordingly. If the duration of the phase was longer than the duration of the observation, we extracted multiple observations from the phase. This however meant that the last observation extracted from a phase was often zero-padded or discarded due to the lack of data. For both WESAD and DTU, there were two phases labeled as a non-stress phase and one as a stress phase, this resulted in the distribution of event and non-event to be around 1/3 event and 2/3 non-event.

5.2.2 Stress Metric: AffectiveROAD

The observations for this dataset were extracted based on the self-reported stress metric. This stress metric was a continuous score between 0 to 1. We standardized the score with a min-max scale of 0 and 1 such that each participant had the same range. For a non-event observation, we required the stress metric to be below the threshold of 0.75. This threshold was selected based on previous research from [Bustos et al. 2021] on the dataset.

Non-event observations were extracted from the data, where the stress metric was always below the threshold of 0.75. For stress observations, the threshold had to be continuously above 0.75. This resulted in a distribution of 2/3 of event segments and 1/3 of non-event segments.

5.2.3 In-the-Wild Datasets: ADARP and Wrist Angel

ADARP and Wrist Angel (WA) are both datasets collected in the wild, where participants used the ”event mark button” on the wristband to tag whenever they felt stressed. Therefore, for these datasets, the observations that were classified as stress (specifically for WA as an OCD event) were data extracted from 1 and 5 minutes, respectively, leading up to the tag. Furthermore, with the ADARP and WA datasets, we experimented with a lead prediction time of 0 to 5 minutes. This means the extracted observations were not leading up to the tag, but ending respectively 0, 1, 2, 3, 4, and 5 minutes before the tag. This enabled us to examine whether we could predict events at a specified time before the event occurred.

For WA, the preprocessing pipeline was heavily inspired by [Lønfeldt et al. 2023]. It required more steps to clean and extract data collected in the wild. The first step was to remove the intervals in which the participant was asleep or did not wear the wristband while data was still being collected.

Second, we created a buffer after a tag which meant no non-event was to be selected within the buffer window, see fig. 5.3.

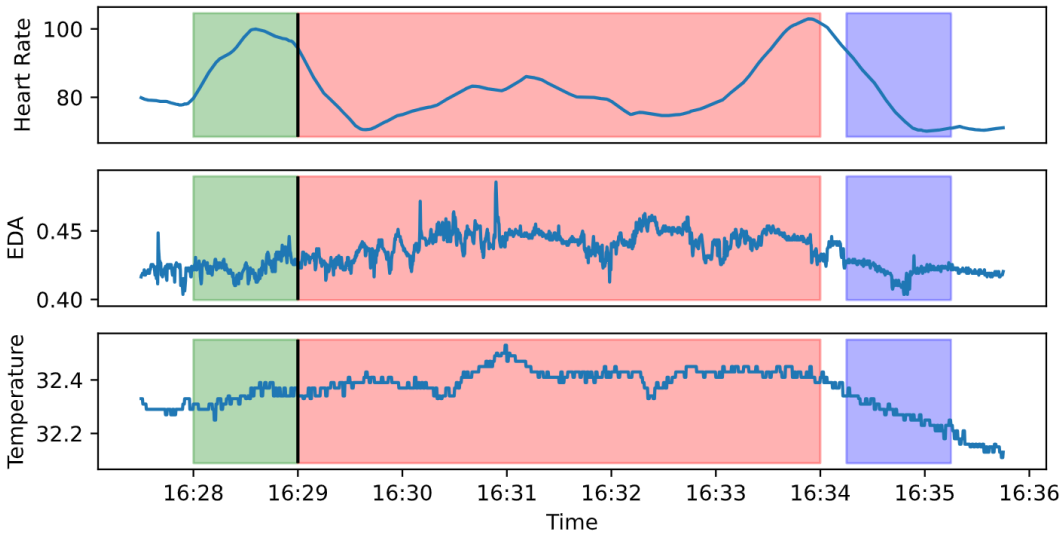


Figure 5.3: Figure taken from [Olesen, Das, et al. 2023] showing a recording from the Wrist Angel dataset. The green area is the observation classified as an event. The solid line is the actual tag. The buffer window is the red area, a specified amount of time immediately after the tag, from which no observations can be selected. The buffer window is five minutes for WA and one hour for ADARP. After the buffer window, the data is again available for randomly selecting non-events (the blue area).

This, however, also removed some tags, and we observed many tags that were within minutes of previous tags, and in these cases, the subsequent tag(s) were ignored. For WA a buffer of 5 minutes was selected based on the previous study [Lønfeldt et al. 2023]. The buffer window for ADARP was significantly higher with a value of 60 minutes selected based on the previous study by [Alinia et al. 2021]. It should be noted that this significantly reduced the number of tags, for ADARP only 32% of the tags generated a valid event segment.

Lastly, since the data was heavily unbalanced in ADARP with 99.3% non-event segments and 0.7% event segments, we applied a random undersampling of the non-event segments. For the Wrist Angel dataset, for each file, we aimed to extract at least 3 non-event segments which was done according to [Lønfeldt et al. 2023]. The distribution for the two classes was kept to approximately 1/3 event segments and 2/3 non-event segments.

5.2.4 Outlier Removal

Due to the nature of in-the-wild recordings, unwanted scenarios happen, such as the wristband not being properly attached to the wrist or not worn at all. To ensure better data quality, we removed segments where it was clear something went wrong. Generally, we searched for an abnormally low average HR, a temperature below 23 degrees, and an absolute average BVP value above 0.35. An example of a removed segment can be seen in fig. 5.4.

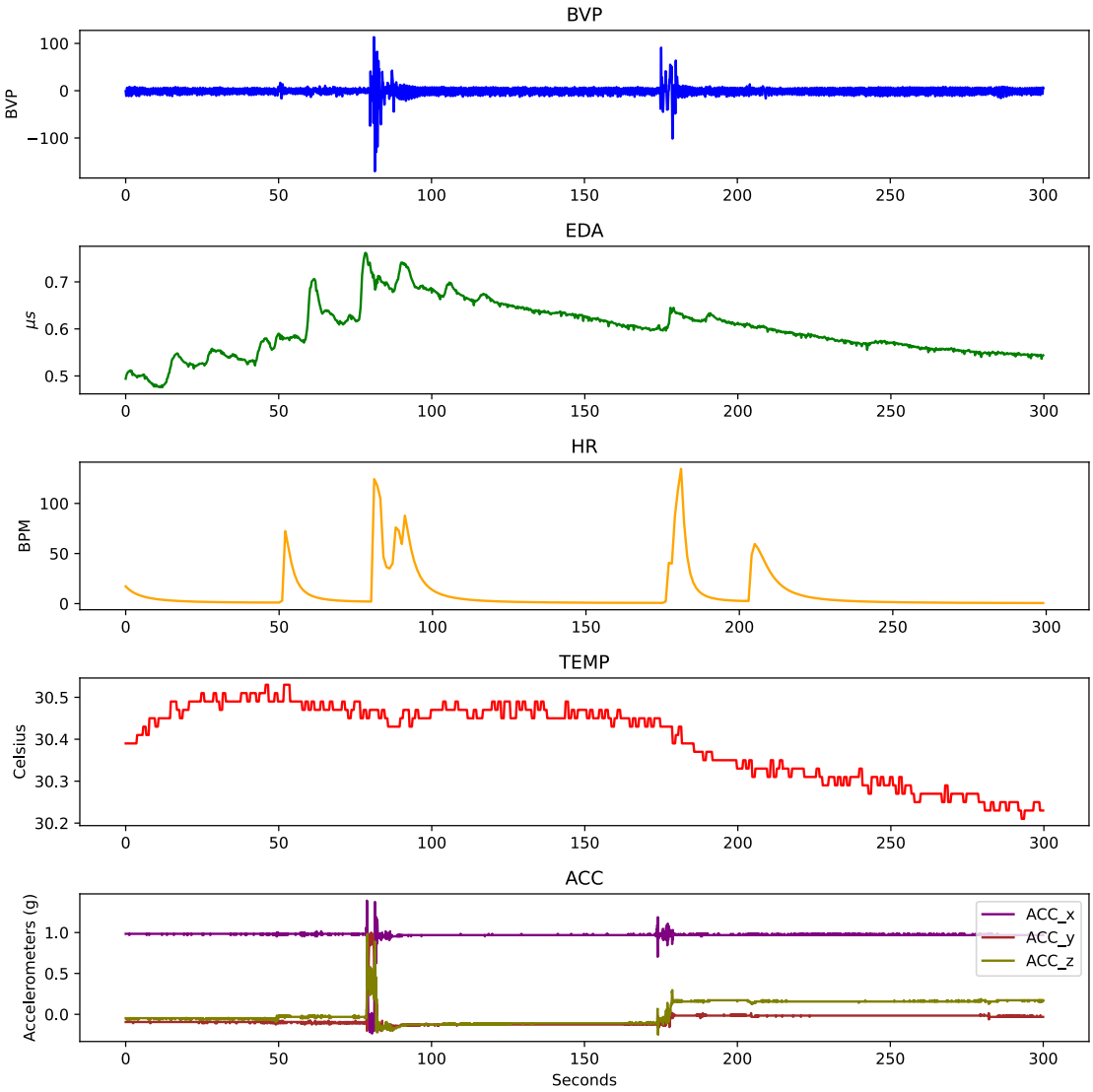


Figure 5.4: An example of how a removed segment can look. This segment is from the WESAD dataset for a non-event segment. The reason for removal was the average HR was 10.31 BPM. This is most likely caused by the wristband not being properly attached.

Furthermore, we decided that if we had less than 10% data for a participant compared to the maximum number of observations for a participant in a dataset, we would remove all segments for that participant. After removing all the outliers, a detailed distribution of events and non-events can be seen in table 5.3.

5 minutes	# event segment	# non-event segment	% event segment
ADARP	106	212	33%
AffectiveRoad	59	24	67%
DTU	104	204	34%
WESAD	29	71	29%
Wrist Angel (WA)	1137	2885	28%
WA lead prediction 1 min.	1074	2602	29%
WA lead prediction 2 min.	1048	2378	31%
WA lead prediction 3 min.	1000	2208	31%
WA lead prediction 4 min.	958	2021	32%
WA lead prediction 5 min.	927	1910	33%
1 minute	# event segment	# non-event segment	% event segment
ADARP	72	144	33%
AffectiveRoad	720	360	71%
DTU	530	1035	34%
WESAD	161	376	30%
Wrist Angel	1446	3547	29%
WA lead prediction 1 min.	1359	3109	30%
WA lead prediction 2 min.	1274	2759	32%
WA lead prediction 3 min.	1181	2477	32%
WA lead prediction 4 min.	1138	2340	33%
WA lead prediction 5 min.	1087	2115	34%

Table 5.3: The distribution of event segments and non-event segments for each dataset. ADARP and Wrist Angel are down-sampled to achieve the distribution shown in the table.

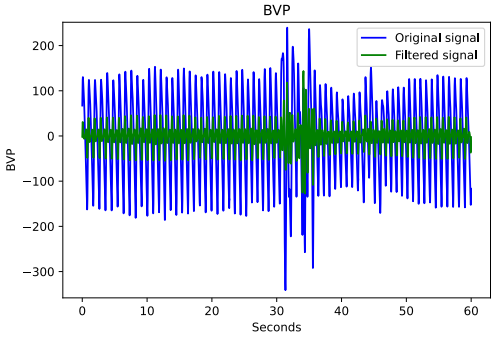
Focusing on the Wrist Angel data with a one-minute window length and zero-minute lead prediction time the amount of sampled event and non-event observations from each participant can be seen in table 5.4. The amount of observations is exclusively from data used in the model, not the entire dataset.

ID	# event segment	# non-event segment	% event segment
3	37	145	20%
4	26	187	12%
5	73	311	23%
6	232	532	44%
7	786	1590	49%
8	275	601	46%

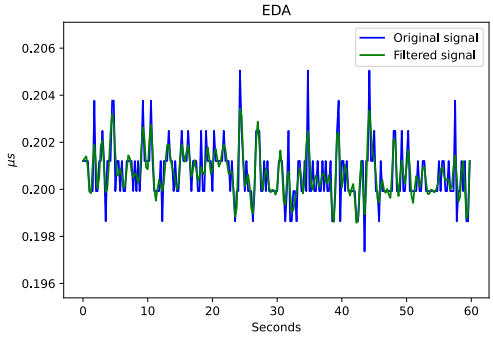
Table 5.4: Observations from participants. The amount of events and non-events has a margin of error of ± 1% due to rounding error. ID 1 and 2, are not included, since they were discarded because of too few event observations. They had 6 and 9 tags respectively according to [Lønfeldt et al. 2023].

5.2.5 Butterworth Filter

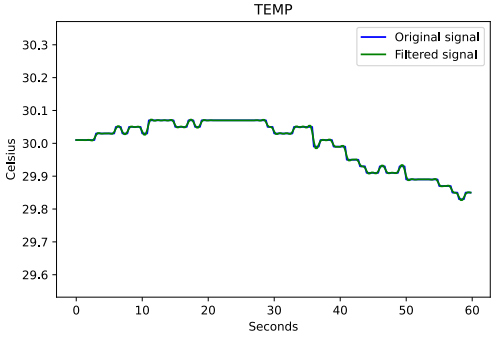
In the wild data contains noise which can negatively affect models. To reduce the noise we used a Butterworth filter. A low-pass Butterworth filter is used to remove high-frequency components. It is done by letting sinusoids with frequencies lower than a specified cut-off frequency go through without any alteration but reducing sinusoids with frequencies greater than the cut-off frequency. The opposite occurs when using a high-pass filter.



(a) Original BVP signal and filtered BVP signal using a $2.$ order band-pass Butterworth filter with frequencies 2Hz and 12Hz.



(b) Original EDA signal and filtered EDA signal with a $6.$ order low-pass Butterworth filter with a cutoff frequency of 1Hz.



(c) Original TEMP signal and filtered TEMP signal with a $6.$ order low-passe Butterworth filter with a cutoff frequency of 1Hz.

Figure 5.5: The signals shown in the graphs are from the ADARP dataset for participant 3 in an event observation. This observation is used to visualize our preprocessing steps. The filter was applied using Python's `scipy` library and the function `scipy.signal.butter`. Note that the HR is not visualized since the Butterworth filter was not applied to this signal.

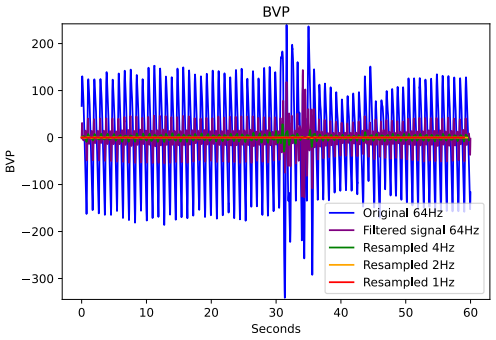
This approach of using a Butterworth filter on BVP, EDA, and TEMP has been widely used in similar studies [Sah et al. 2022; Alinia et al. 2021; Lopez-Martinez, El-Haouij, and Picard 2019; W. Chen, Zheng, and Sun 2021; Schmidt et al. 2018]. In our final design, we selected to use a $6.$ order low-pass Butterworth filter with a cut-off frequency of 1Hz for TEMP and EDA signals based on [Lønfeldt et al. 2023]. Furthermore, inspired by Lønfeldt et al. we used a $2.$ order band-pass filter with a low-pass cut-off frequency of 2Hz and a high-pass cut-off frequency of 12Hz for filtering BVP signals selected based on results from section 6.4. An example of a filtered observation can be seen in fig. 5.5.

5.2.6 Frequency Resampling

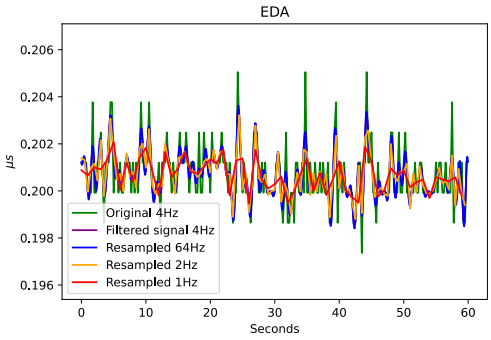
The signals we selected to use for the prediction of stress are BVP, EDA, HR, and temperature. However, these signals are captured at different sampling rates. To be able to obtain the same dimension for each of the signals, we needed to resample. For our four signals, we used two different resampling methods. The first was resampling using Python's SciPy library [Virtanen et al. 2020], more specifically `scipy.signal.resample`. This function re-samples the data using a Fourier method to a given number of samples. This function uses a fast Fourier transform (FFT) to resample a 1D signal. For the signals

BVP and EDA, we selected this sampling method.

The second resampling method was linear interpolation using Python's NumPy library [Harris et al. 2020] and its `numpy.interp` function. This method was used for the temperature and HR signals. As seen in fig. 5.6 temperature and HR are both similar to linear lines, thus making this method more suited.

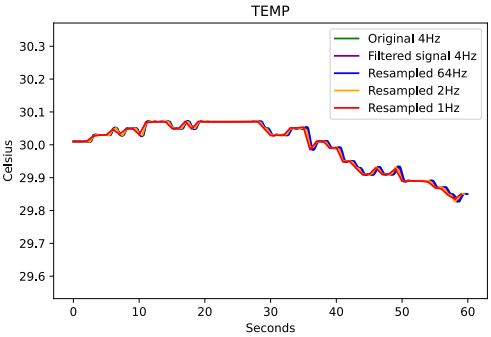


(a) Raw and resampled signal of BVP using `scipy.signal.resample`.



(b) Raw and resmapled signal of EDA using `scipy.signal.resample`. The purple line filtered signal at 4Hz is covered by the blue resampled signal at 64Hz.

(c) Raw and resmapled signal of heart rate using `numpy.interp`.



(d) Raw and resmapled signal of temperature using `numpy.interp`.

Figure 5.6: Plots of the four signals: BVP, EDA, HR, and TEMP. Each plot contains the original signal, the filtered signal, and the resampled signals based on the filtered signal. In each subplot, four different sampling frequencies are shown. 64Hz is the highest sampling frequency we have from the E4 wristband for BVP. 4Hz which is the original sampling frequency for EDA and TEMP. 2Hz to test if we can downsample more than 4Hz. Finally, 1Hz is the sampling frequency for HR. From the plot, we see that HR and TEMP are not affected by up- or downsampling, as they are in BVP and EDA. For BVP we see that any downsampling will distort the signal, but 4Hz still seems to keep the original wavelength, however, the amplitude is lowered. For EDA an up-sampling to 64Hz will mimic the 4Hz original signal but will introduce redundancy.

Mainly four different sampling frequencies have been tested: 1Hz, 2Hz, 4Hz, and 64Hz, which can be seen in figure fig. 5.6. When downsampling is applied, we will inevitability distort the signal, while upsampling will create redundancy in data meaning needing more computational power for deep learning models.

The final design is inspired by [Deznabi and Fiterau 2023] where we used all four signals

in 4Hz and 64Hz. These frequencies are selected based on visual inspection and the results of the exploratory analysis in section 6.2.

5.2.7 Standardization

Two different standardization methods were used: by the training set or by observation. The two methods standardize each signal individually, therefore transforming the data from each signal to have a mean of 0 and a standard deviation of 1 [Jaadi 2024].

Training

The data is standardized by mean and standard deviation obtained from the training dataset:

$$X_{d,s} = \frac{X_{d,s} - \mu_{\text{train},s}}{\sigma_{\text{train},s}}, \begin{array}{l} d \in [\text{train, validation, test}] \\ s \in [\text{BVP, EDA, HR, TEMP}] \end{array} \tag{5.1}$$

Where $X_{d,s}$ is the data for a given physiological signal, $\mu_{train,s}$ is the mean value of the training data ($X_{train,s}$), and $\sigma_{train,s}$ is the standard deviation of the training data for the given signal. Standardizing data by the training dataset is a widely used and simple method, which is why this was the method we used in the initial tests. Furthermore, we used this method when we were only dealing with one dataset and not multiple, since there were differences between the datasets.

Observation Wise

When including multiple datasets, we generally standardized by the mean and standard deviation of each observation:

$$X_{\text{obs},s} = \frac{X_{\text{obs},s} - u_{\text{obs},s}}{\sigma_{\text{obs},s}}, \begin{array}{l} \text{obs} \in [0, 1, ..., n] \\ s \in [\text{BVP, EDA, HR, TEMP}] \end{array} \tag{5.2}$$

Where $X_{\text{obs},s}$ is the observation and physiological signal to standardize, $\mu_{\text{obs},s}$ is the mean of the observation ($X_{\text{obs},s}$) and $\sigma_{\text{obs},s}$ is the standard deviation of the observation for the given signal.

Thus, standardizing by observation results in each observation and physiological signal having a mean of 0 and a standard deviation of 1. While if standardizing by the entire training dataset, the individual observations can vary a lot in mean and standard deviation.

A plot of an observation after standardizing can be seen in appendix A.11.

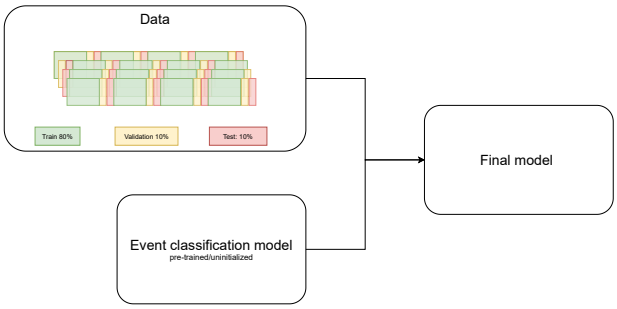
5.2.8 Data Splits

Two different data splits were used for pre-training and fine-tuning. For pre-training an 80-20 training and validation split was used [Rácz, Bajusz, and Héberger 2021]. In this step, we trained on 80% of the data and validated on the remaining 20%. It is to be noted that a separate dataset was used as the test set, where either the Wrist Angel dataset or another separate dataset for testing purposes.

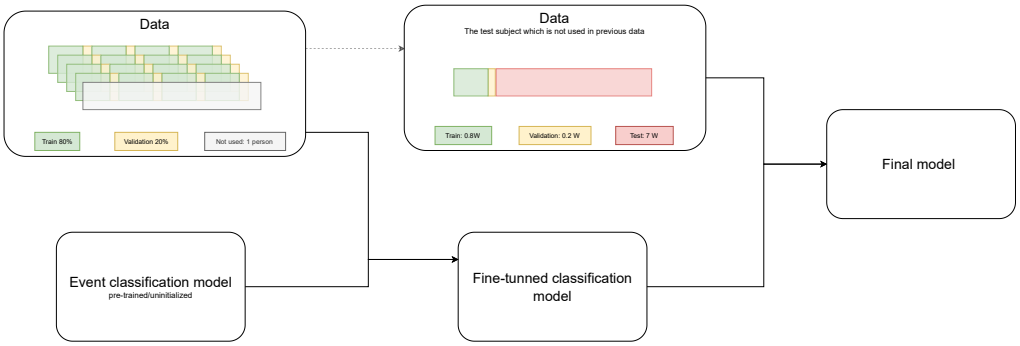
Random Split

The test dataset was used for fine-tuning where the split was 80-10-10 for training, validation, and test sets, respectively. To achieve a random split the `numpy.random.shuffle` function was used, and with a seed to ensure reproducibility. Additionally, it was ensured that each person was present in each of the training, validation, and test datasets.

Personalized Split
Here, one participant is extracted as the test set. The remaining participant's data are divided into 80% training data and 20% validation data, which was one training loop. The test participant is then split into a training and validation set, with the first week of data collected in an 80-20 split. The remaining weeks were then used as the test dataset, for a second training loop. Both of these methods of splitting data for fine-tuning are visualized in fig. 5.7.



(a) Flow for data split method: Random.



(b) Flow for data split method: Random + Personalized.

Figure 5.7: Visualization of the two data splitting methods for fine-tuning and training. The figures are from our statistical analysis plan [Skat-Rørdam et al. 2023] as seen in appendix A.19.

5.3 Evaluation Methods

We used accuracy and F1-score [B 2019] to evaluate and assess the performance of our model and different applications. The accuracy and F1-score are given by eqs. (5.3) and (5.4).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{5.3}$$

$$F1\text{-}score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{5.4}$$

Where TP is true positives, TN is true negatives, FP is false positives, and FN is false negatives.

Accuracy is the overall percentage of correct classifications. The main reason we use accuracy is due to comparability with other papers and studies because accuracy is one

of the most used evaluation metrics [Hossin and Sulaiman 2015]. Accuracy does not work well for imbalanced data, and given none of the datasets used were balanced, we also include the F1-score as an evaluation metric. The F1 score is the harmonic mean of the two metrics $Precision$ and $Recall$, which each penalizes false positives and false negatives, respectively. Hence, it combines these two and therefore works well when data is imbalanced.

5.4 Statistical tests

5.4.1 Mann-Whitney U rank test

The Mann-Whitney U rank test is a non-parametric test of the null hypothesis H_0: That two populations are equal [McClenaghan 2024].

The advantage of non-parametric statistical tests such as the Mann-Whitney U rank test is that it does not assume data to be normally distributed, which is relevant since we are looking at small sample sizes of only 10 (the number of runs in one scenario) [Taylor 2024]. It is assumed that the two datasets for comparison have a similar shape and samples are independent. The independent assumption is a disadvantage due to the different WA datasets originating from the same data. An alternative test for classification is McNemar's test, but this was in most cases not an option, since we test across multiple datasets (with different lengths and data points), hence the observations are not comparable between tested populations.

The Mann-Whitney test uses the U-statistic, which is defined as the minimum of U_1 and U_2, calculated as seen in eq. (5.5) [McClenaghan 2024].

$$U_1 = n_1 n_2 + \frac{n_1 (n_1 + 1)}{2} - R_1, U_2 = n_1 n_2 + \frac{n_2 (n_2 + 1)}{2} - R_2 \tag{5.5}$$

Here n_1 and n_2 are the sample sizes of the two samples, respectively. R_1 and R_2 are the sum of the ranks of the two samples, respectively. Where the ranks are at which number each value from the samples comes in an ascending ordered list.

The U-statistic can then be compared to a critical value or used to calculate a p-value for comparison with a significance level.

In practice, the U-statistic and p-value from the Mann-Whitney U rank test can be computed with the `scipy.stats.mannwhitneyu` function in Python.

5.4.2 McNemar's test

The McNemar's test [McNemar 1947] is a statistical test for paired data. Within machine learning it can be used to compare the performance of two classification models tested on the same dataset. Hence the null hypothesis is then H_0: The performance of the two models is equal.

McNemar's test uses a χ^2 test statistic which is calculated as in eq. (5.6).

$$\chi^2 = \frac{(b - c)^2}{(b + c)} \tag{5.6}$$

Or more often used is the continuity corrected version as eq. (5.7) [Edwards 1948].

$$\chi^2 = \frac{(|b - c| - 1)^2}{(b + c)} \tag{5.7}$$

Where b and c comes from a $2x2$ contingency table with the predictions of the two models as seen in fig. 5.8 from [Raschka 2024].

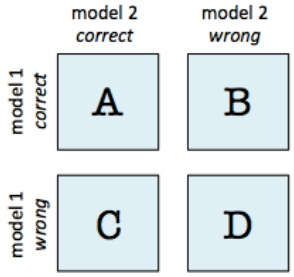


Figure 5.8: Contingency table for McNemar's test used to compare models. From [Raschka 2024].

In practice the two functions `mcnemar_table` and `mcnemar` from [Raschka 2018] are used with the predictions and true labels to produce a p-value and determine significance.

5.4.3 Benjamini-Hochberg Correction

When performing a statistical test, the risk of a false rejected null hypothesis, H_0 (type I error) is the significance level, α, where $\alpha = 0.05$ is typically used, given a risk of 5%. However, when multiple tests are performed, the expected number of false rejections will increase. The Benjamini-Hochberg correction controls for this False Discovery Rate (FDR), which is the expected proportion of false rejections among all rejections [S. Chen, Feng, and Yi 2017]. To accomplish this all null hypotheses are first sorted in ascending order based on their p-values. Then the index k is found as the largest index (i), which satisfies eq. (5.8).

$$P_{(i)} \leq \frac{i}{m} q \tag{5.8}$$

Where $P_{(i)}$ is the p-values at index i, m is the total amount of tests made, and q is the specified upper bound of the FDR, typically 0.05. Then all hypotheses $H_{(i)} i = 1, 2, ..., k$ can be rejected [Benjamini and Hochberg 1995].

In practice, we have used the Python function `statsmodels.stats.multitest.multipletests` to adjust the p-values based on the Benjamini-Hochberg correction.

5.5 Model Exploration and Selection

To find a suitable model efficiently, model selection and testing have been carried out throughout this project simultaneously with the discovery of data(sets) and the specification of run conditions. This means the selection should be interpreted as two phases: an exploratory phase in which the changes were broader and the comparisons more generalized. And a model selection phase wherein a model architecture was chosen, but with more rigorous testing.

In the exploratory phase, we tried two different frameworks, PyTorch [Paszke et al. 2019] and TensorFlow [Martín Abadi et al. 2015] with Keras [Chollet et al. 2015]. For each framework, we explored multiple models on different datasets to find the best framework and model for our use case. The Keras framework was only used in the initial phase of this project, hence PyTorch was the primary framework, and the only one used on the

Wrist Angel dataset, which is why all deep learning presented in this thesis is conducted in PyTorch.

Given that the physiological signals were sampled in different frequencies the resulting input for the model will also be of different lengths. As mentioned earlier in section 5.2.6 this can be accounted for by up- or downsampling. However, the results of each method require distortion of the data, which can worsen the performance. To account for this, we included a model for each sample frequency that was specified to be included. This is done by stacking a model for each sample frequency and then combining the results of each model at the end (see fig. 5.9). The idea of combining models for different frequencies comes from [Deznabi and Fiterau 2023], which is executed in the PyTorch framework.

5.5.1 Model Exploration

In the exploratory phase, 7 model architectures were tested: LSTM, Transformer, CNN + Attention, CNN + LSTM, FCN, FCN per channel, and CNN + Attention per channel. The reason why these seven models were tested is that the code we have used is inspired by [Deznabi and Fiterau 2023], and these models were part of their code. Each architecture only applies to the encoder, as the decoder and classifier heads both are linear layers. The linear layers are used instead of more complex layers to reduce the model complexity.

LSTM
Long short-term memory (LSTM) is a type of recurrent neural network (RNN) that aims to prevent the problem of vanishing gradients. The LSTM network is typically used in sequential data problems. An LSTM unit typically consists of 4 parts. A cell that stores memory, this memory can be over an arbitrary time interval, a forget gate that can throw away unnecessary information, an input gate that instead preserves information, and an output gate to manipulate which memory to use. [NVIDIA 2024a]

Transformer
Transformers are mainly used in large language models (LLMs) because they can learn the context between sequential data. To achieve this, transformers use a multi-head attention mechanism. Attention calculates a weight for each input in a given context, which allows it to detect more distant influences, compared to a typical RNN. [Merritt 2022]

CNN with Attention
A convolution neural network (CNN) typically consists of three layers: an input, a hidden, and an output layer. These layers typically contain pooling and normalization. [NVIDIA 2024b]. The CNN is explained in more detail in section 5.5.2. The attention layer is added to the output layer and helps to detect distant influences, as explained above for the transformer model.

CNN with LSTM
The CNN with LSTM model is similar to the CNN with Attention model as described previously, but without any attention and with an additional LSTM layer at the end of the model.

FCN
A fully convolutional network (FCN) is similar to the CNN models as described above but without any added attention, LSTM layer, or similar. Hence the FCN only consists of the convolutional layers, including normalization and pooling.

FCN per Channel
A model similar to the FCN described above, but in which each physiological signal is treated separately. This means that one model is created for each signal; hence, this method creates multiple models, all with only one input feature.

CNN with Attention per Channel
This method also creates a model per signal similar to the FCN per channel model. However, here the CNN with Attention is used for each of the models instead of an FCN.

5.5.2 Model Selection - Convolutional Neural Network (CNN)

The final model which was the one used on the Wrist Angel dataset was a CNN. This model is described in more detail in this section. Convolutional Neural Networks are best known for their superior performance in image recognition but have also been proven to perform well in time series classification [Zhao et al. 2017]. A CNN typically consists of three types of layers: The convolutional layer, the pooling layer, and the fully connected layer [IBM 2023], where the convolutional layer is the primary part of a CNN. In this layer, a small matrix, called a kernel, slides over the data and calculates the dot product between the kernel and the input data. This process is called convolution and maps the input data to a feature map. After that, the pooling layer down-samples the data, also by filtering with a kernel, but instead of using weights, it aggregates the data to the output matrix. The pooling is performed by computing the average or max value, where max pooling is the most common [IBM 2023], and also what we have used in this thesis.

The specific architecture of the CNN model that we have used is shown in fig. 5.9.

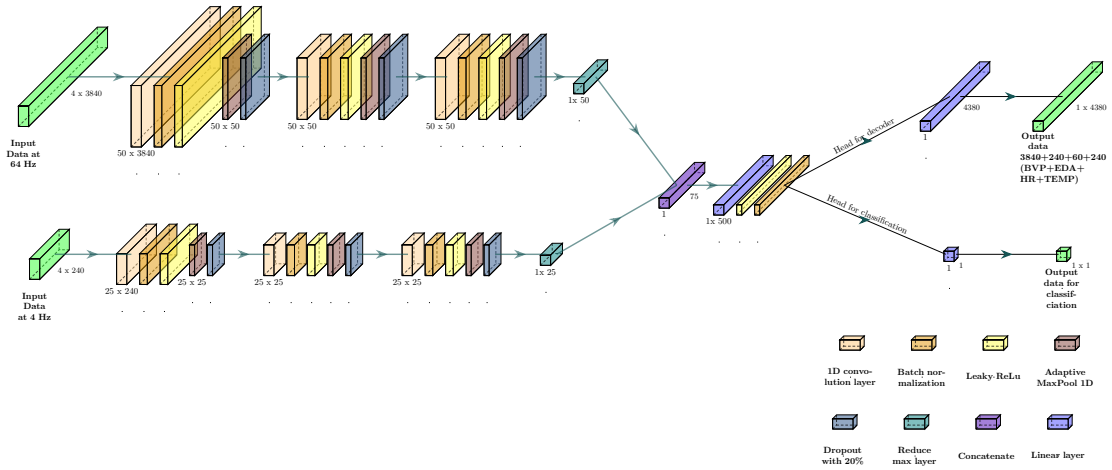


Figure 5.9: Visualisation of the CNN model. The model is shown for 1-minute data. See appendix A.12 for a full-size figure. The model takes input in both 4Hz and 64Hz, and each frequency has 3 convolutional layers. Thereafter by concatenating the two frequencies, we obtain our bottleneck of 75 nodes. In the figure, we see a split at the end, where the one pointing upwards is for the decoder and the one pointing downward is for the classification head.

As can be seen from fig. 5.9 the model takes each physiological signal as input for given sample rates. In this case, for 1-minute observations, the sample rates are 4Hz ($60s \cdot 4Hz = 240$ data points) and 64Hz ($60s \cdot 64Hz = 3840$ data points) with a batch size of 64, and the 4 physiological signals (BVP, EDA, HR, and TEMP). By including both 4Hz and 64Hz, the BVP, EDA, and TEMP data will be used in their original frequencies. The intuition is that down- or upsampling, either loses or creates data, thus hurting the performance. Therefore, we intended to keep most data in its original frequency. Another approach could have been to stack the different signals in their original frequency, and only in their original frequency, meaning 1 signal for 1HZ pipeline, 2 for 4HZ pipeline, and 1 for 64Hz pipeline. However, this would result in the lack of temporal knowledge

and correlation between the signals, as signals are only summed up in the end. For that reason, we kept the signals together in the same up or down-sampled frequencies such that the temporal aspect and signal correlations were kept.

For both frequency models, the input signal is transposed from
[batch size, signal length, # signals] $\rightarrow$ [batch size, # signals, signal length]. E.g. for the 4Hz model in fig. 5.9: $[64, 240, 4] \rightarrow [64, 4, 240]$. This is done because the input for the convolutional block is N, C_{in}, L_{in} where N is the batch size, C is the number of channels/variables (in our case, the physiological signals), and L is the length of the signal sequence per [PyTorch 2023]. The first layer is then a convolutional layer. For each convolutional layer, the following steps occur in order:

1. After a convolutional layer, a 1D batch normalization is applied. It is applied to increase numerical stability and works by normalizing each feature per batch, thus batch normalization. It is defined as in eq. (5.9) from [PyTorch 2024b].

$$y = \frac{x - \mathsf{E}[x]}{\sqrt{\mathsf{Var}[x] + \epsilon}}\gamma + \beta \tag{5.9}$$

   Additionally, scale, γ, and shift, β, are added to ensure the identity transform. These parameters are learned along with conjunction. Finally, the error term ϵ is added for numerical stability [Ioffe and Szegedy 2015].

2. Then the activation function is added. We use the Leaky Rectified Linear Unit (LReLU), which is defined in eq. (5.10) from [PyTorch 2024d]:

$$\mathsf{LeakyReLU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \mathsf{negative\_slope} \times x, & \text{otherwise} \end{cases} \tag{5.10}$$

   It leaves any positive value as is, but applies a negative slope to any negative value. The default negative slope in PyTorch is 0.01, which is what we used. This is different from ReLU, which sets any negative value to 0. ReLU is the most commonly used activation in CNNs [Potrimba 2023] given its low computation cost and not having the vanishing gradients problem, unlike TanH and Sigmoid. The advantage of LReLU over ReLU is that it ensures the contribution of all neurons, even though they are negative, hence the network does not become too sparse [Rallabandi 2023].

3. Thereafter a max pooling layer is applied. Max pooling is used to reduce the spatial dimensions of the data, not the dimensionality. It is done by taking the maximum of a given window (kernel), such that only the most important features are used. In our case `ADAPTIVEMAXPOOL1D` [PyTorch 2024a] is used. For this function, only the output size needs to be specified, thus simplifying the code.

4. Finally, a dropout of 0.2 is applied as this prevents co-adaption of neurons [PyTorch 2024c].

Each of the four steps is repeated after all convolutional layers, which in this case is 3 times. Finally, for each frequency model, the outputs of the layers are averaged to obtain the dimensionality of the specified output nodes for the given frequency. As can be seen from fig. 5.9, it will be 50 and 25 nodes. The output of the two pipelines is then combined

by stacking them on top of each other. Then two linear layers are applied, with LReLU and batch norm being applied after the first linear layer. The autoencoder and classification models differ in the output dimensions of the linear layers. In the autoencoder, the dimensions are $75 \rightarrow 500 \rightarrow signal\ length$, thus reconstructing the signals. For classification, the dimensions are $75 \rightarrow 500 \rightarrow 1$. Notice no sigmoid function is used, thus this has to be applied afterward to get probabilities of an event.

5.6 Transfer Learning

As mentioned in section 4.3 the Wrist Angel dataset only consists of data from nine participants. Consequently, we had a data scarcity problem. To mitigate the problem, additional training steps were implemented before the final application of the model on the Wrist Angel dataset.

5.6.1 Pre-Training and Fine-Tuning

First, an autoencoder was trained to reconstruct the four original physiological signals. This initialized features, with the hope that the model would learn the physiological signals. These features could then later be used for the prediction task. The idea was that it would be easier for a classification model to detect stress based on the encoded features, rather than training with an uninitialized network. Second, the decoder was replaced with a classification head, with the task being to classify stress. With these two steps, a classification model was trained on the open source datasets (see section 4.4) to predict stress events. With the expectation (see chapter 3) that stress events look physiologically similar to OCD events, which were to be detected in the Wrist Angel data.

Finally, the pre-trained classification model could then be used to directly classify or to be fine-tuned on the Wrist Angel dataset.

5.6.2 Self-Supervised Learning

Different self-supervised learning tasks were implemented - specifically three augmentation tasks and a contrastive task. The three different augmentation types tested were: additive Gaussian noise, 0-masking, and amplitude scaling. In each instance, the pre-trained autoencoder was used, where the target was the reconstruction of the original signals. The contrastive task was to determine if the two input observations belonged to the same person. For contrastive learning, the encoder part of the autoencoder was used, with an added contrastive head on top, like in the classification model. Here the output dimension was a vector space with a length equal to the number of unique participants in the dataset, instead of 1 like the classification model. A positive pair was defined as two observations belonging to the same participant, with a negative pair being observations from two different participants. To calculate the loss between pairs the `BYOLoss` [Grill et al. 2020; Adaloglou 2022] loss function was used.

5.7 Activity Classification

Physical activity can look similar to acute stress on a person's physiological signals. Therefore, we used a basic classifier to distinguish between physiological signals indicative of physical activity. This classifier was based on accelerometer data collected from the participants. When using the physical activity classification, we initially filtered out observations identified as physical activity from the dataset. The remaining data, presumed to represent non-physical activity states, underwent fine-tuning, training, and prediction. On the other hand, in scenarios where we omitted the physical activity classification, observations were still assessed using the physical activity classifier, but the data remained unfiltered. These predictions were stored for comparison with our OCD predictions, as shown in section 6.9.2.

To get an initial understanding of the accelerometer data for resting and physical activity, we examined various random observations, such as the example seen in fig. 4.8. This led us to examine two different classifiers, both based on a threshold. One for the overall standard deviation of the acceleration, which we call the ”Standard Deviation Threshold”, and one for the most prominent frequency, which we call the ”FFT Frequency Threshold”.

To find the appropriate thresholds we used the resting and running observations from the WEEE dataset (section 4.4.5). We ignored the cycling observations because arm movements are very limited while biking (see fig. 4.7). Furthermore, we also tested on the WESAD dataset (section 4.4.4), where we assumed all observations to be non-physical-activity based on the experiment design. This gave us a total of 163 observations. 31 physical activity observations from the WEEE dataset and $32 + 100 =$132 observations from respectively WEEE and WESAD datasets.

After finding a threshold for the open-source data we further evaluated it on a separate dataset collected in a controlled physical activity session under the same study as Wrist Angel. However, the data does not come from the same participants as in our OCD WA dataset, meaning the data is collected from a control group of adolescents without OCD. During this process, we first used thresholds found based on WEEE and WESAD and evaluated the performance. Thereafter, we decided if we could use one of the two methods with the pre-defined threshold or if we would need to find a new threshold based on the new dataset, due to the difference between adults and adolescents.

Standard Deviation Threshold
The standard deviation threshold is the standard deviation of the acceleration in all three directions for the observation. The threshold was found by calculating the minimum and maximum standard deviation for respectively resting observations and physical activity observations. Then a histogram of the standard deviations was plotted, to show the difference in standard deviation distribution between resting and physical activity. The plot is used to visually identify a threshold separating resting and activity. The threshold was selected with a low tolerance for false positives.

FFT Frequency Threshold
Based on the signals visualized in figs. 4.7 and 4.8 we decided to investigate the signals in the frequency domain to examine if there was some identifiable difference between rest and activity observation. Thus the method FFT frequency threshold was applied. We computed the discrete Fourier transform and its sample frequencies of each observation using `scipy.fft.rfft` and `scipy.fft.rfftfreq`. With these functions, we created a visualization of the amplitude (y-axis) by frequency (x-axis). Based on the investigation we determined the direction(s) of the accelerometer data which were useful for selecting a threshold or a range if needed.

5.8 Application of Model on Wrist Angel Data

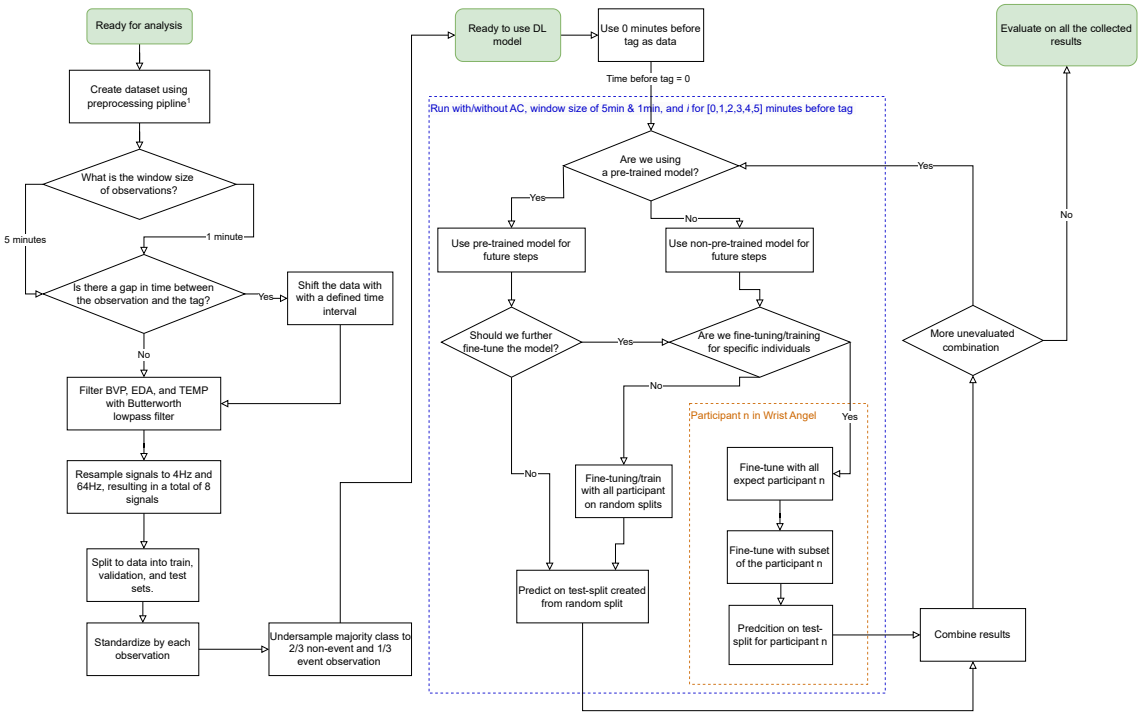


Figure 5.10: A flow chart describing the steps in signal processing and data analysis used for the WA dataset. The flow chart is a modified version of figure 1 in [Skat-Rørdam et al. 2023]. AC = activity classification.
[1] The preprocessing pipeline is the same as used in [Lønfeldt et al. 2023].

As described in our statistical analysis plan [Skat-Rørdam et al. 2023] (see appendix A.19) and visualized in fig. 5.10, we applied our event classification model in the following five different ways.

1. We classified the Wrist Angel data directly with our pre-trained model, which was neither trained nor fine-tuned on any Wrist Angel data. Hence the entire Wrist Angel dataset was the test set for our model.
2. We fine-tuned our pre-trained model to part of the Wrist Angel data and used the rest as test data. We fine-tuned in the following two different ways, to assess whether fine-tuning improves the performance of our pre-trained model and whether personalized fine-tuning is advantageous.
   (a) Random (fig. 5.7a): We randomly split the Wrist Angel data in 80% training data, 10% validation data and 10% test data.
   (b) Random + Personalized (fig. 5.7b): A combination of a random fine-tuning and a personalized fine-tuning. First, one participant was extracted as the test person. The remaining 7 participants were randomly split into 80% training data and 20% validation data for the first fine-tuning. The second fine-tuning was performed on the extracted test participant. The first week was randomly divided into 80% training data and 20% validation data. Finally, data from the remaining 7 weeks of the test participant were used as test data.
3. Training our uninitialized model from scratch with the Wrist Angel data. Hence, we

did not use the pre-trained model. The training was done with the same two different procedures as explained with the fine-tuning, see fig. 5.7.

Besides applying our model in each of these five different ways, we applied it using data of window length 1 minute and 5 minutes, to see if the window length affected the performance. Furthermore, we applied our model with and without an initial classification of physical activity as described in section 5.7. Finally, for each of these scenarios, we also used six different lead prediction times of 0, 1, 2, 3, 4, and 5 minutes. Lead prediction time (LPT) means the time between the end of the signal and the actual OCD event tag. Hence, the amount of time ahead of an OCD event we can predict it.

In total, this gave us 120 different scenarios for applying our model to the WristAngel data.

$$\text{Number of scenarios} = 5 \cdot 2 \cdot 2 \cdot 6 = 120 \tag{5.11}$$

Each of these 120 scenarios was run approximately 10 times. The non-personalized scenarios (direct classification, fine-tuning with a random split, and training from scratch with a random split) were run exactly 10 times. The personalized scenarios (fine-tuning and training) were run 2 times for each participant belonging to the test set. Having six participants with enough data for personal fine-tuning/training, we thereby got 12 runs for each personalized scenario.

6 Results

This chapter focuses on results from the PyTorch framework. First, we present various results from exploring different models, section 6.1. Hereafter follows a section about finding the best CNN model, tuning hyperparameters, and data parameters, section 6.2. Then there are the final results from applying our model to the Wrist Angel data, section 6.8. Finally, in section 6.9 we present different analyses of the Wrist Angel results. Even though the results were sub-par, we extracted insights and learnings about the data and the prediction task. These will be further elaborated in the discussion in chapter 7.

6.1 Model Exploration

As mentioned in section 5.5, we tested both the PyTorch and Keras frameworks to create our deep learning models. We ended up using PyTorch, mainly because of the flexibility, compatibility, and a small performance advantage. The results of some selected models in Keras can be seen in appendix A.13.

After having decided to use PyTorch over Keras, we explored different model architectures and types of models.

Initially, we tested seven different model architectures, which were available in the code of [Deznabi and Fiterau 2023] as described in section 5.5.1. All these models were tested once on the DTU 1-min dataset.

	Accuracy	F1-score	Run time
LSTM	Train: 0.34 Val: 0.38 Test: 0.33	Train 0.51 Val 0.55 Test 0.49	20m9s
Transformer	Train: 0.36 Val: 0.4 Test: 0.39	Train: 0.52 Val: 0.56 Test: 0.52	24m11s
CNN with Attention	Train: 0.47 Val: 0.46 Test: 0.49	Train: 0.56 Val: 0.57 Test: 0.56	**2m10s**
CNN with LSTM	Train: 0.38 Val: 0.44 Test: 0.38	Train: 0.52 Val: 0.57 Test: 0.51	9m32s
FCN	Train: 0.54 Val: 0.52 **Test: 0.58**	Train: 0.6 Val: 0.61 **Test: 0.61**	3m33s
FCN + channel	Train: 0.38 Val: 0.44 Test: 0.38	Train: 0.53 Val: 0.57 Test: 0.51	13m12s
CNN + Attention, per channel	**Train: 0.56** **Val: 0.55** Test: 0.55	**Train: 0.61** **Val: 0.62** Test: 0.59	4m13s

Table 6.1: Accuracy and F1-scores of different model architectures. Each model was run once on the DTU 1-min dataset. Bold numbers are the best performance for respectively train, val, and test across all models. Furthermore, for run time the fastest model is bolded.

From table 6.1 it is clear that more complex models, such as the transformer and the LSTM take significantly longer to run. However, this added complexity does not improve performance, it negatively impacts it. The best models for our task seem to be FCN and CNN + Attention, per channel. Furthermore, CNN + Attention is the fastest model to run, and the performance is not much worse than the best models.

Moving on with CNN + Attention, FCN, and CNN + Attention per channel, we tested these three architectures on the DTU 6-min dataset as seen in table 6.2.

Model	Accuracy	F1-score	Run time
CNN + Attention	Train: 0.77 Val: 0.88 Test: 0.68	Train: 0.74 Val: 0.88 **Test: 0.62**	**2m19s**
FCN	**Train: 0.92** **Val: 0.91** **Test: 0.74**	**Train: 0.89** **Val: 0.9** Test: 0.6	4m0s
CNN + Attention per channel	Train: 0.56 Val: 0.55 Test: 0.55	Train: 0.61 Val: 0.62 Test: 0.59	4m13s

Table 6.2: Accuracy and F1-scores of different model architectures. Each model was run once on the DTU 6-min dataset. Bold numbers are the best performance for respectively train, val, and test across all models. Furthermore, for run time the fastest model is bolded.

.

FCN performs the best of the three in table 6.2, but CNN + Attention is faster to run. For this reason, we continued testing both different CNN and FCN models on the DTU+WESAD 5-min datasets. These results can be seen in table 6.3 and additional FCN model tests for the DTU+WESAD 5-min dataset can be seen in appendix A.15. Furthermore, we tested additional FCN architectures on the DTU+WESAD 1-min dataset, these can be seen in appendix A.14.

Model	Accuracy	F1-score
CNN + Attention BN: 4Hz 50, 64Hz 25	**Train: 1.0** **Val: 0.81** Test: 0.58	**Train: 1.0** **Val: 0.77** Test: 0.52
CNN + Attention BN: 4Hz 50, 64Hz 10	Train: 0.95 Val: 0.77 Test: 0.61	Train: 0.93 Val: 0.74 Test: 0.6
Best FCN out of 20 different architectures See all FCN tests in appendix A.15	Train: 0.79 Val: 0.77 **Test: 0.71**	Train: 0.73 Val: 0.72 **Test: 0.69**

Table 6.3: Accuracy and F1-score of different model architectures. Each model was run once on the DTU+WESAD 5-min dataset. Bold numbers are the best performance for respectively train, val, and test across all models. BN: The number of nodes in the bottleneck of the models.

From table 6.3 CNN + Attention seems to do the best in training and validation, hence perhaps overfitting, whereas FCN performs the best on the test split. However, we have continued developing the CNN model. The architecture is selected through an overall

evaluation of the performance of the CNN + Attention models and the FCN models. In addition, the fact that CNN + Attention is a lot faster to run than the FCN model. Nonetheless, FCN and CNN + Attention are similar.

6.2 Initial Model Testing

After deciding to use a CNN model in the exploratory phase, we continued testing different CNN model architectures for the final selection. We tested different model parameters such as number of nodes in the bottleneck, kernel sizes, max pooling, the number of layers, the structure of the network, the inclusion of attention, and self-supervised learning techniques.

Bottleneck per Frequency

First, different amounts of nodes per frequency in the bottleneck were tested. It was initially tested for a CNN + Attention model on the DTU 4-min dataset in table 6.4. After that, it was tested for a regular CNN model on the ADARP 5-min dataset in table 6.5.

Nodes in bottleneck	Accuracy	F1-score
4Hz: 50 64Hz: 25	**Train: 0.98** **Val: 0.78** Test: 0.59	**Train: 0.98** **Val: 0.70** Test: 0.48
4Hz: 100 64Hz: 50	Train: 0.32 Val: 0.38 Test: 0.34	Train: 0.48 Val: 0.52 **Test: 0.51**
4Hz: 10 64Hz: 5	Train: 0.68 Val: 0.62 **Test: 0.66**	Train: 0.00 Val: 0.00 Test: 0.00
4Hz: 25 64Hz: 5	Train: 0.91 Val: 0.69 Test: 0.53	Train: 0.88 Val: 0.55 Test: 0.35

Table 6.4: Accuracy and F1-score for a different amount of nodes in the bottleneck for the 4Hz and 64Hz models. Each model was run once on the DTU 4-min dataset. Bold numbers are the best performance for respectively train, val, and test across all models.

Nodes in bottleneck	Accuracy μ	F1-score μ	Run time μ
4Hz: 50	Train: 0.69 Val: 0.61 Test: 0.57	Train: 0.65 Val: 0.55 Test: 0.55	Not known
64Hz: 50	Train: 0.69 Val: 0.66 Test: 0.57	Train: 0.64 Val: 0.58 Test: 0.59	1-min38s
4Hz: 25 64Hz: 50	**Train: 0.79** Val: 0.73 **Test: 0.64**	**Train: 0.72** Val: 0.67 Test: 0.54	1-min38s
4Hz: 50 64Hz: 25	Train: 0.76 **Val: 0.77** Test: 0.63	Train: 0.71 **Val: 0.71** **Test: 0.60**	**1-min20s**

Table 6.5: Average accuracy and F1-score over three runs for different numbers of nodes in the bottleneck of the CNN model. The test was performed on the ADARP 5-min dataset. Bold numbers are the best performance for respectively train, val, and test across all models. Furthermore, for run time the fastest model is bolded.

From table 6.4 it can be seen that 50 nodes in the 4Hz model, and 25 nodes in the 64Hz model seem to perform the best. Increasing the complexity in row two to 100 and 50 nodes generally does not seem to improve performance. Neither does simplifying the model. When it becomes too simplified to 10 and 5 nodes, it even predicts everything as non-stress, resulting in an F1-score of 0. Furthermore, from table 6.5 we see an indication that the combination of a 4Hz and a 64Hz model performs better than only using one down or up-sampled model. Whether there are 50 nodes in one model and 25 in the other or the opposite does not seem to matter much, but since 50 nodes in 4Hz and 25 nodes in 64Hz are best in validation, and take less time to run, we decided to continue with this combination.

Kernel Sizes
Different kernel sizes were also experimented with for 10-layer networks. A comparison of these can be seen in table 6.6.

Kernel sizes	Accuracy	F1-score
[3,5,5,7,9,7,5,5,3,3]	Train: 0.98 **Val: 0.78** Test: 0.59	Train: 0.98 **Val: 0.70** Test: 0.48
$\frac{[9,7,5,3,3,3,3,5,7,9]}{3}$	Train: 0.67 Val: 0.62 Test: 0.53	Train: 0.47 Val: 0.40 Test: 0.12
[3,3,3,3,3,3,3,3,3,3]	**Train: 1.00** Val: 0.69 Test: 0.62	**Train: 0.99** Val: 0.55 Test: 0.40
[9,9,9,9,9,9,9,9,9,9]	Train: 0.55 Val: 0.47 **Test: 0.69**	Train: 0.40 Val: 0.37 **Test: 0.58**
[1,1,1,1,1,1,1,1,1,1]	Train: 0.60 Val: 0.53 Test: 0.41	Train: 0.55 Val: 0.48 Test: 0.17

Table 6.6: Accuracy and F1-score for different kernel sizes for the CNN + Attention model. Each model was run once on the DTU 4-min dataset. Bold numbers are the best performance for respectively train, val, and test across all models.

In table 6.16 no large difference in performance is seen between the first three rows. To simplify we decided to continue with a kernel size of 3 for each layer. In later tests (see section 6.4) we return to testing other kernel sizes.

Max Pooling
We also tested a few different max pooling kernel sizes on the DTU 4-min dataset as seen in table 6.7.

CNN + Attention Model	Accuracy	F1-score
Max pooling kernel size = [3,5,5,7,9,7,5,5,3,3]	Train: 0.64 Val: 0.53 **Test: 0.56**	Train: 0.43 Val: 0.44 Test: 0.36
Max pooling kernel size = [3,3,3,3,3,3,3,3,3,3]	**Train: 0.88** **Val: 0.66** **Test: 0.56**	**Train: 0.84** **Val: 0.62** **Test: 0.42**
No Max Pooling	Train: 0.42 Val: 0.41 Test: 0.41	Train: 0.5 Val: 0.46 **Test: 0.42**

Table 6.7: Accuracy and F1-score of different max pooling kernel sizes. Each run once on the DTU 4-min dataset. Bold numbers are the best performance for respectively train, val, and test across all models.

Here, we see that max pooling with a kernel size of 3 is the best option. Therefore, this is what we have used for future tests. Until we use the adaptive max pooling technique instead (see table 6.19) to limit the number of hyperparameters.

Number of Layers
Below in table 6.8, we tested different amounts of layers in the network. One test on a uniform network with a kernel size of 3, and one test on a reverse U-Net[1] network with a kernel size of 5.

Uniform, kernel size: 3				Reverse U-Net, kernel size: 5			
Number of layers	Accuracy μ	F1-score μ	Run time μ	Number of layers	Accuracy μ	F1-score μ	Run time μ
3	**Train: 0.76** **Val: 0.77** **Test: 0.63**	**Train: 0.71** **Val: 0.71** **Test: 0.60**	1-min20s	3	**Train: 0.73** **Val: 0.71** **Test: 0.54**	**Train: 0.67** **Val: 0.63** **Test: 0.57**	**1-min1s**
10	Train: 0.67 Val: 0.68 **Test: 0.63**	Train: 0.55 Val: 0.53 Test: 0.59	1-min20s	5	**Train: 0.73** Val: 0.66 Test: 0.48	Train: 0.66 Val: 0.57 Test: 0.39	1-min7s

Table 6.8: Average accuracy and F1-score over 3 runs for a different number of layers. The test was carried out on the ADARP 5-minute dataset. Bold numbers are the best performance for respectively train, val, and test across all models. Furthermore, for run time the fastest model is bolded.

Looking at table 6.8 we see no evidence that the model's performance is increased by adding more than 3 layers.

[1]Which is a usual U-Net [Ronneberger, Fischer, and Brox 2015], but reversed; hence first an expansive path followed by a contracting path.

Structure of Network
Additionally, we tested both a uniform structure and a reverse U-Net structure. This was done for both 10 and 5-layer networks, with a kernel size of 3 and 5. These tests can be seen in table 6.9.

10 layers, kernel size: 3				5 layers, kernel size: 5			
Structure	Accuracy μ	F1-score μ	Run time μ	Structure	Accuracy μ	F1-score μ	Run time μ
Uniform	Train: 0.67 **Val: 0.68** **Test: 0.63**	Train: 0.55 **Val: 0.53** **Test: 0.59**	**1-min20s**	Uniform	Train: 0.70 **Val: 0.66** **Test: 0.52**	Train: 0.63 Val: 0.56 **Test: 0.49**	**56s**
Reverse U-Net	**Train: 0.77** Val: 0.56 Test: 0.51	**Train: 0.71** Val: 0.47 Test: 0.49	1-min53s	Reverse U-Net	**Train: 0.73** **Val: 0.66** Test: 0.48	**Train: 0.66** **Val: 0.57** Test: 0.39	1-min7s

Table 6.9: Average accuracy and F1-score over 3 runs for different network structures. The test was carried out on the ADARP 5-min dataset. Bold numbers are the best performance for respectively train, val, and test across models with the same number of layers. Furthermore, for run time the fastest model is bolded.

From table 6.9 it can be seen that the reverse U-Net does not improve performance, and the uniform structure is faster to run.

Attention Mechanism
Then we tested if the attention mechanism improved the model in table 6.10. This test was performed for both a 3-layer uniform network and a 10-layer reverse U-Net network.

3 layers, uniform				10 layers, reverse U-Net			
Attention	Accuracy μ	F1-score μ	Run time μ	Attention	Accuracy μ	F1-score μ	Run time μ
No	**Train: 0.76** **Val: 0.77** **Test: 0.63**	**Train: 0.71** **Val: 0.71** **Test: 0.60**	1-min20s	No	**Train: 0.77** **Val: 0.56** **Test: 0.51**	**Train: 0.71** Val: 0.47 Test: 0.49	**1-min53s**
Yes	Train: 0.71 Val: 0.58 Test: 0.49	**Train: 0.71** Val: 0.55 Test: 0.50	**54s**	Yes	Train: 0.52 Val: 0.47 Test: 0.48	Train: 0.55 **Val: 0.50** **Test: 0.52**	3min50s

Table 6.10: Average accuracy and F1-score over 3 runs with and without attention. The test was performed on the ADARP 5-min dataset. Bold numbers are the best performance for respectively train, val, and test across models with the same number of layers. Furthermore, for run time the fastest model is bolded.

In table 6.10 it can be seen that attention does not improve performance, especially not for the 3-layer uniform network, which so far seems to perform the best.

Self-Supervised Learning
In addition to testing different model parameters, we also experimented with some self-supervised learning. These tests can be seen in table 6.11. The tests were done on the best-performing model obtained at this point. Hence for the number of nodes in the bottleneck, we used 50 nodes in the 4Hz model, and 25 nodes in the 64Hz model (table 6.5). 3 layers were used according to table 6.8. We used a uniform structure due to table 6.9, and a kernel size of 3 based on table 6.6. Finally, attention was not included (table 6.10). These tests were also done on the ADARP 5-min dataset, similar to tables 6.8 to 6.10.

Self supervised method	Accuracy μ	F1-score μ	Run time μ
No self supervised learning	Train: 0.73 Val: 0.69 **Test: 0.74**	Train: 0.64 Val: 0.61 **Test: 0.70**	1-min2s
No self supervised learning No Autoencoder	Train: 0.73 **Val: 0.78** Test: 0.41	Train: 0.65 **Val: 0.73** Test: 0.43	**29s**
Self masking	**Train: 0.79** Val: 0.66 Test: 0.58	**Train: 0.72** Val: 0.59 Test: 049	1-min
Self noise	Train: 0.71 Val: 0.61 Test: 0.65	Train: 0.66 Val: 0.59 Test: 0.63	1-min4s
Self scaling (/100)	Train: 0.75 Val: 0.64 Test: 0.54	Train: 0.69 Val: 0.59 Test: 0.52	1-min32s
Self person	**Train: 0.79** Val: 0.73 Test: 0.60	Train: 0.70 Val: 0.66 Test: 0.55	1-min11s

Table 6.11: Average accuracy and F1-score over 3 runs for different self-supervised learning methods. The test was carried out on the ADARP 5-min dataset. Bold numbers are the best performance for respectively train, val, and test across all pretext tasks. Furthermore, for run time the fastest model is bolded.

In table 6.11 we see that different self-supervised learning methods are best for training, validation, and testing, respectively. Therefore, there is no clear support for self-supervised learning that increases performance. This is why we have decided to keep the model simple, and not include any self-supervised learning in our final models. Interestingly, the model in row two, with no self-supervised learning, nor an autoencoder, performs quite well in training and validation compared to the other models. However, it does seem to be overfitting more than if the autoencoder is included.

6.3 Three CNN Model Comparisons

After testing different CNN models as seen above in tables 6.4 to 6.11 we chose three different architectures for more thorough testing: simple, medium and complex architecture with specifications seen in table 6.12.

Model name	Attention	4Hz bottleneck	64Hz bottleneck	Number of hidden layers	Structure of hidden layers	Extra linear layer (500 nodes)
Simple	No	10	-	3	Uniform	No
Medium	No	50	25	3	Uniform	Yes
Complex	Yes	50	25	10	Reverse U-Net	Yes

Table 6.12: Overview of model specifications for the simple, medium, and complex model used in the thorough test in table 6.13.

The simple model was chosen, because we have seen overfitting, and therefore wanted to simplify the model to reduce overfitting and increase the performance for the test split. The medium model is the model we would expect to perform the best, based on our analysis of the different model parameters for the CNN model (section 6.2). Because, from tables 6.4

and 6.5, we find a combination of a 4Hz model with 50 nodes in the bottleneck and a 64Hz model with 25 nodes in the bottleneck to perform the best. From tables 6.8 to 6.10 we find 3 hidden layers in a uniform structure, with no attention included to be best. The complex model was also tested to see if it could improve performance when including attention and increasing the complexity of the model regarding depth.

Each model was tested leaving out one dataset as a test dataset and using the remaining three datasets as training and validation datasets. Each test has been performed 10 times, and the average and standard deviation of the accuracy and F1-score of these tests are reported in table 6.13.

Test Dataset:		ADARP		DTU	
Model		Accuracy	F1-score	Accuracy	F1-score
Simple	Train	0.47 (0.07)	0.44 (0.21)	0.52 (0.05)	0.38 (0.22)
	Val	0.46 (0.10)	0.41 (0.18)	0.54 (0.12)	0.34 (0.20)
	Test	0.46 (0.16)	0.34 (0.20)	0.51 (0.10)	0.30 (0.20)
Medium	Train	**0.80 (0.02)**	**0.78 (0.02)**	**0.76 (0.03)**	**0.71 (0.04)**
	Val	**0.73 (0.02)**	**0.64 (0.02)**	**0.72 (0.03)**	**0.56 (0.05)**
	Test	0.44 **(0.02)**	**0.47 (0.06)**	**0.59 (0.01)**	**0.37 (0.03)**
Complex	Train	0.58 (0.09)	0.57 (0.08)	0.51 (0.07)	0.41 (0.18)
	Val	0.58 (0.06)	0.51 (0.07)	0.51 (0.09)	0.34 (0.17)
	Test	**0.50** (0.09)	0.39 (0.15)	0.52 (0.08)	0.33 (0.17)

Test Dataset:		ROAD		WESAD	
Model		Accuracy	F1-score	Accuracy	F1-score
Simple	Train	0.52 (0.13)	0.32 (0.16)	0.49 (0.08)	0.37 (0.20)
	Val	0.51 (0.10)	0.34 (0.19)	0.49 (0.06)	0.41 (0.21)
	Test	0.46 (0.17)	0.42 (0.33)	0.51 (0.18)	0.39 (0.14)
Medium	Train	**0.58 (0.04)**	**0.55 (0.02)**	**0.67 (0.04)**	**0.60 (0.04)**
	Val	**0.61 (0.03)**	**0.60 (0.03)**	**0.59 (0.03)**	**0.54 (0.04)**
	Test	**0.65 (0.04)**	**0.76 (0.03)**	**0.71 (0.04)**	0.15 **(0.10)**
Complex	Train	0.52 (0.9)	0.37 (0.12)	0.50 (0.06)	0.36 (0.23)
	Val	0.52 (0.08)	0.41 (0.16)	0.51 (0.06)	0.40 (0.28)
	Test	0.52 (0.09)	0.53 (0.28)	0.54 (0.12)	**0.32** (0.18)

Table 6.13: Average accuracy and F1-score of 10 runs with standard deviation reported in parenthesis. Each of the three CNN models was tested on each dataset with a 5-minute window length, where the remaining three datasets were used as training and validation datasets. From this table, it is seen that the medium complexity model performs best. Bold numbers are the best performance for respectively train, val, and test across all models for each dataset, and the bold numbers in the parentheses indicate the smallest standard deviation.

As seen in table 6.13 the model with medium complexity architecture is, as expected superior, as it performs better in almost all tests. Therefore, we have continued with the medium complexity model for some final testing.

6.4 Final Model Testing

After selecting the medium complexity model from table 6.13 as our final model, we tested the data preprocessing techniques and hyperparameters of the model. For all the following tests it applies that if one dataset is used as the test dataset, the remaining three datasets are used to train the pre-trained model, and the fine-tuning is done on the test dataset. Additionally, all datasets use a 5-minute window length. Furthermore, the evaluation metrics are only reported for the test dataset, since this is what we are mostly interested in improving. Because the test dataset is unseen data, and hence the best representation of a real-world situation, where an OCD event is to be predicted from unknown real-time data. Additionally, this was the case with the Wrist Angel data being unseen.

Standardization

First, we tested different standardization methods as seen below in table 6.14.

	Pre-trained F1-score		Fine-tuned F1-score	
Standardization	μ	σ	μ	σ
All signals per observation	**0.37**	0.04	**0.60**	**0.03**
All signals per training data	0.36	0.06	0.43	0.13
Only BVP + EDA per obs	0.36	**0.02**	0.54	0.09

Table 6.14: Average and standard deviation F1-score for DTU as the test dataset. Each standardization method was run 10 times. Bold numbers are the best performance across all methods.

From the standardization test presented in table 6.14, the best standardization method is to standardize all signals per observation. This is especially the case for fine-tuning, as for pre-training, there are no significant differences in performance.

Butterworth Filter for BVP

As mentioned in section 5.2.5 we used a 6th order low pass Butterworth filter with a cutoff frequency of 1Hz for EDA and TEMP. However, for BVP we tested different Butterworth filters, which are presented in tables 6.15 and A.6. The first test was performed on the DTU dataset as the test dataset and the data is standardized per observation.

Butterworth filter			F1-score	
Highcut	Lowcut	Order	μ	σ
12	2	2	**0.56**	0.07
14	2	1	0.55	**0.06**
14	2	2	0.54	0.08
Nabian filter			0.52	0.13

Table 6.15: Average and standard deviation F1-score after fine-tuning on DTU as the test dataset. Each filter was run 10 times. Data was standardized per observation. The Nabian filter comes from the NeuroKit function `ppg_clean` with `method='nabian2018'`. A plot of an example of the outcome of the different filters can be seen in appendix A.10. Bold numbers are the best performance for the different filters.

The test on the DTU dataset (table 6.15) shows that the second-order Butterworth filter with highcut 12Hz and lowcut 2Hz performs best, but the improvement does not seem to be significant. Additional tests were made with ADARP as the test dataset; here the results were similar. These tests can be seen in appendix A.16.

Kernel Size
With the chosen medium complexity model, we also tested three different kernel sizes on the ADARP dataset. The test can be seen in table 6.16.

	Pre-trained F1-score		Fine-tuned F1-score	
Kernel size	μ	σ	μ	σ
[3,3,3]	0.50	0.03	**0.65**	**0.04**
[3,5,7]	**0.52**	**0.01**	0.55	0.09
[7,5,3]	0.50	0.03	0.57	0.07

Table 6.16: Average and standard deviation F1-score for ADARP as the test dataset. Each kernel size was run 10 times. Standardized per observation and using the Butterworth filter, highcut: 12Hz, lowcut: 2Hz, order: 2. Bold numbers are the best performance across the different kernel sizes.

Since using a kernel size of 3 is best after fine-tuning according to the test in table 6.16, this is what we used for the Wrist Angel dataset.

Number of Layers
Additionally, we again tested different numbers of layers, respectively 1, 3, and 5 layers. But neither decreasing to 1 nor increasing the number of layers to 5 seems to improve the performance compared to the 3 layers, which were previously used. The test is presented in table 6.17.

	Pre-trained F1-score		Fine-tuned F1-score	
Number of layers	μ	σ	μ	σ
1	**0.51**	**0.01**	0.63	0.09
3	0.50	0.03	**0.65**	**0.04**
5	**0.51**	**0.01**	0.63	0.05

Table 6.17: Average and standard deviation F1-score for ADARP as the test dataset. Each number of layers was run 10 times. Standardized per observation and using the Butterworth filter, highcut: 12, lowcut: 2, order: 2. Bold numbers are the best performance across different numbers of layers.

Attention Mechanism
With this medium complexity model, we again tested whether attention improves the model. In table 6.18 we see the pre-training improves a little bit with attention, but the fine-tuning is best without attention. Hence, we still did not find evidence of attention improving the model, which is why we did not include it in the final model.

	Pre-trained F1-score		Fine-tuned F1-score	
Attention	μ	σ	μ	σ
With Attention	**0.51**	**0.02**	0.61	0.07
Without Attention	0.50	0.03	**0.65**	**0.04**

Table 6.18: Average and standard deviation F1-score for ADARP as the test dataset. Each test was run 10 times. Standardized per observation and using Butterworth filter, high-cut: 12Hz, low-cut: 2Hz, order: 2. Bold numbers are the best performance for comparing with or without attention mechanism.

LReLU and Adaptive Max Pool
On the DTU dataset, we also tested different combinations of using the LReLU activation function and adaptive max pooling.

		Pre-trained F1-score		Fine-tuned F1-score	
LReLU	Max Pool	μ	σ	μ	σ
No	No	**0.47**	**0.03**	0.50	0.06
Yes	No	0.46	0.07	0.54	0.07
Yes	Adaptive	0.43	0.07	**0.64**	**0.03**
Yes	MP1D	0.42	0.06	**0.64**	0.05

Table 6.19: Average and standard deviation F1-score of 10 runs for pre-trained and fine-tuned. Tested on the DTU dataset. MP1D is `Maxpool1D`. Bold numbers are the best performance across the different combinations.

It can be seen that not including any activation function nor adaptive max pooling is best for the pre-training, but when it comes to fine-tuning, it is best to include both the LReLU activation function and max pooling. There is no difference in F1-score between the two max pooling types, however, the adaptive max pooling is approximately 50% faster. Therefore, we used the adaptive max pooling as well as the LReLU activation function in the final model to be used on the Wrist Angel dataset.

Lead Prediction Time (LPT)
We tested fine-tuning with multiple lead prediction times to see if it was possible to predict a stress event sometime before it happened. The results are presented in table 6.20.

	1-min Data, F1-score		5-min Data, F1-score	
	μ	σ	μ	σ
Pre-trained baseline model	0.38	0.05	0.50	0.03
Lead Prediction Time for Fine-tune				
0min (baseline)	0.46	0.23	0.59	0.06
1-min	0.39	0.10	0.56	0.08
2-min	**0.53**	**0.07**	**0.60**	**0.04**
3-min	0.46	0.09	0.45	0.08
4-min	0.44	0.08	0.45	0.06
5-min	0.51	0.14	0.55	0.10

Table 6.20: Average and standard deviation F1-score for ADARP as the test dataset, respectively as 1-min and 5-min data. Each test was run 10 times. The pre-trained baseline model is trained on DTU, ROAD, and WESAD with no lead prediction time. Bold numbers are the best performance across all LPTs.

From table 6.20 we see that the performance is best when the lead prediction time is 2-min. This is interesting since that would mean it would be easier to predict a stress event 2 minutes before it happens rather than just before it happens. Since this result is counterintuitive, we have decided to test the lead prediction time again in our final test on the Wrist Angel data. These results can be seen in section 6.8.2.

Optimizer
Finally, we tested different optimizers. The results are presented in table 6.21. The model was trained on all data, fine-tuned, and tested on WESAD. Each test was run 10 times,

and the reported F1-scores are an average and standard deviation of the 10 runs for the test dataset.

	F1-score	
Optimizer	μ	σ
Adam	**0.89**	**0.10**
Nadam	0.88	0.11
AMSGrad	0.82	0.15

Table 6.21: Average and standard deviation F1-score of 10 runs, with varying optimizers. Reported for WESAD as the test dataset. 5-minute data was used. Bold numbers are the best performance across the different optimizers.

From table 6.21 it can be seen that there is not that big a difference between the optimizers, but Adam performs the best. Hence we used Adam as our optimizer when applying the model on the Wrist Angel dataset.

6.5 Simulated Data

As mentioned in section 4.6 we simulated data, with the hopes that adding simulated data would improve the model's performance since the amount of data is increased. Data was simulated with two different methods (see section 5.1): The plain method, and the fragmented method adding more noise.

In table 6.22 the effect of including the plain simulated data can be seen. This test was done on our "medium" complexity model, before choosing adaptive max pooling (table 6.19), trained on the WESAD, ROAD, and ADARP datasets, and tested on the DTU dataset.

	Without	Plain
Pre-training F1 test score	**0.37 (0.04)**	**0.37** (0.05)
Fine-tuned F1 test score	**0.6 (0.03)**	0.53 (0.10)

Table 6.22: Average F1-score of 10 runs for the DTU 5-min dataset as test dataset. The standard deviation is given in parentheses. Bold numbers are the best performance between not including and including plain simulated data.

As seen in table 6.22 the addition of the plain simulated data did not improve the pre-training phase and additionally worsened the performance when comparing fine-tuned results. Additionally, the noisy simulated data (created with the fragmented method) was tested early in our exploratory phase, where it also only worsened the performance of the model. Therefore, we did not include simulated data when pre-training our final model.

However, towards the end of this project, we tested the simulated data once again, but this time with our final model. These results are shown in table 6.23 and corresponding McNemar's tests between the methods in table 6.24.

		Simulation method	
	No simulation	Plain	Fragmented
Test dataset	F1-score	F1-score	F1-score
ADARP	0.52 **(0.09)**	0.48 (0.18)	**0.59** (0.17)
DTU	0.60 **(0.04)**	0.54 (0.07)	**0.73** (0.05)
ROAD	**0.79** (0.05)	0.70 **(0.04)**	0.72 (0.06)
WESAD	0.74 (0.15)	0.84 **(0.04)**	**0.93** (0.12)

Table 6.23: Average F1-score over 10 runs. Fine-tuned using the random split and tested on each dataset with the remaining three datasets used to pre-train the model. 5-minute data is used. The standard deviation is given in parentheses. Bold numbers are the best performance for each dataset, and the bold numbers in the parentheses indicate the smallest standard deviation.

Test	ADARP	DTU	ROAD	WESAD
No Sim v. Plain Sim	0.141	**0.024**	0.682	0.898
No Sim v. Fragmented Sim	**0.018**	**3E-04**	0.640	0.078
Plain Sim v. Fragmented Sim	**3E-05**	**2E-09**	1.0	0.219

Table 6.24: P-values for McNemar's test between including different simulated data or not. All p-values are adjusted according to the Benjamini-Hochberg correction with 12 tests. Bold p-values are significant on a 5% significance level.

After testing the simulated data, we see in tables 6.23 and 6.24 that the plain simulated data generally still does not improve performance. For the DTU dataset the F1-score significantly decreased when the plain simulated data was included. However, we now see that the simulated data created with the fragmented method increases the performance of the model in all datasets, except for the ROAD dataset. The increase in performance is only significant for ADARP and DTU, but ADARP is notable since it is an in-the-wild dataset as the Wrist Angel dataset.[2]

[2]Given these results, it would have been interesting to include the simulated data when testing on the Wrist Angel dataset, but this was not possible due to time restrictions.

6.6 Evaluation of Pre-Trained Model on Open-Source Datasets

The following results are obtained by applying the final model we found in section 6.4. Furthermore, hyperparameters used can be seen in table 6.26.

Dataset	Scenario	F1-score
ADARP	Predict	**0.30** (0.05)
	Pre-trained random	0.25 (**0.00**)
	Train random	0.26 (0.10)
DTU	Predict	**0.57** (**0.01**)
	Pre-trained random	0.48 (0.07)
	Train random	0.53 (0.05)
ROAD	Predict	0.58 (**0.01**)
	Pre-trained random	**0.67** (0.07)
	Train random	0.55 (0.08)
WESAD	Predict	0.90 (0.02)
	Pre-trained random	**0.95** (**0.01**)
	Train random	0.89 (0.04)

Table 6.25: Average F1-score over 10 runs. Tested on each dataset and different applications of the model with 1-minute data. The standard deviation is given in parentheses. Bold numbers are the best performance for each dataset across the different scenarios, and the bold numbers in the parentheses indicate the smallest standard deviation.

From table 6.25 we see that for the datasets ROAD and WESAD, the best performance is obtained by fine-tuning the pre-trained model. However, for datasets ADARP and DTU, the best performance is not obtained by training the model on the dataset in any way (either fine-tuning or directly training), but instead by using the pre-trained model to directly predict. Given that performance is worsened by the model learning and training on the data, this indicates the model has trouble learning these datasets, most likely due to their complexity.

6.7 Physical Activity Model

In this section, we will examine the performance of our physical activity classification for the standard deviation threshold method and FFT Frequency Threshold method. For both methods, we will first examine how the threshold was found. Next, we will examine the performance of this pre-defined threshold on the unseen adolescent data collected during the Wrist Angel study from a control group. Based on the performance we will determine which of the methods will be selected as the final physical activity classification model.

6.7.1 Standard Deviation Threshold

In fig. 6.1 a histogram of the standard deviation for our open-source datasets is plotted. The intent is to be able to visually identify differences in the distributions of rest and physical activity. From this histogram (fig. 6.1), we see to some extent a cut in the distributions. The minimum standard deviation of a physical activity observation is 33.8 and the maximum standard deviation from a resting observation is 38.1. Hence, as seen in the histogram, there is some overlap. Prioritizing zero false positives, we chose the threshold to be 40. This gave us the confusion matrix seen in fig. 6.2, an accuracy of 0.95, and an F1-score of 0.85 for the 132 observations from WEEE and WESAD.

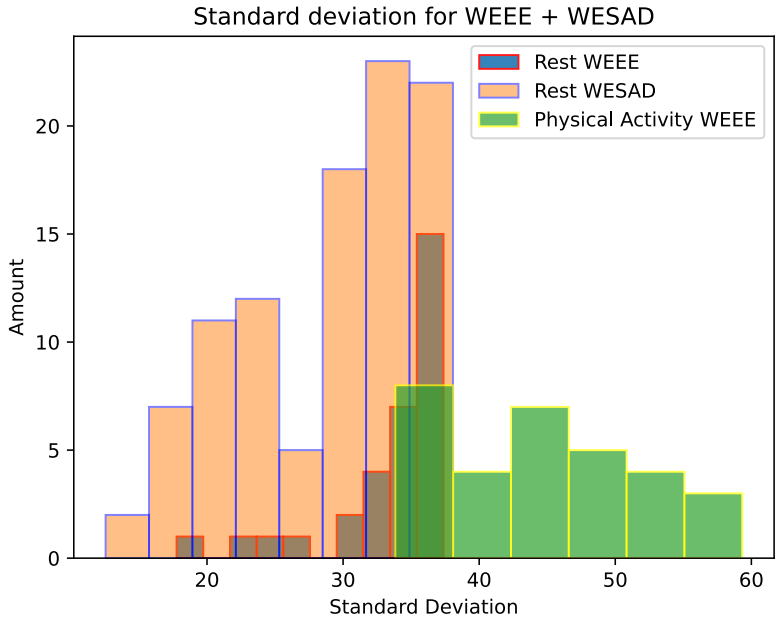


Figure 6.1: Histogram of standard deviations for resting and physical activity observations. All physical activity observation comes from the WEEE dataset. It is seen that no resting observations have a standard deviation above 40.

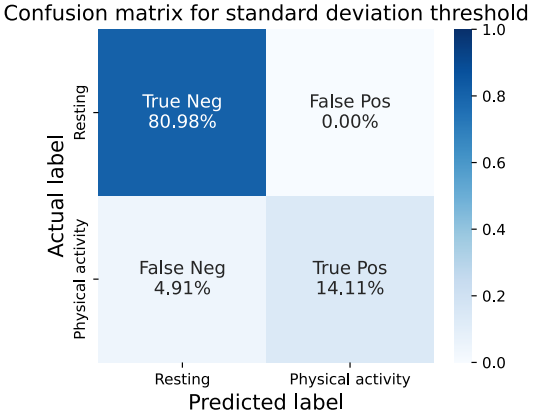
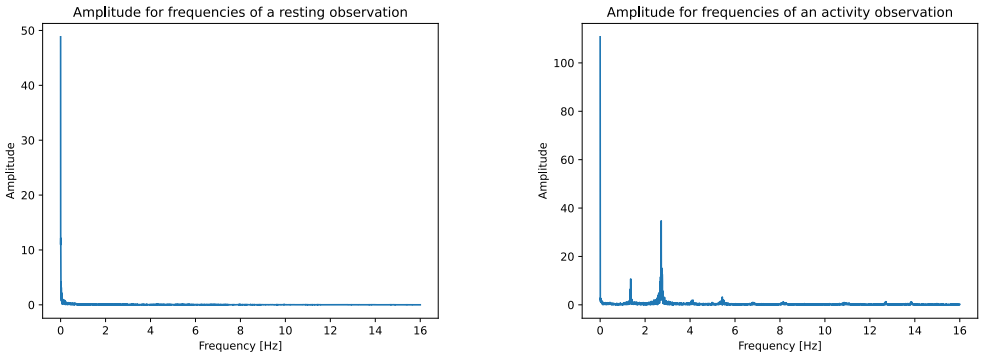


Figure 6.2: Confusion matrix of physical activity classification with standard deviation threshold of 40. The data evaluated are from the WEEE and WESAD datasets.

6.7.2 Fast Fourier Transform (FFT) Frequency Threshold

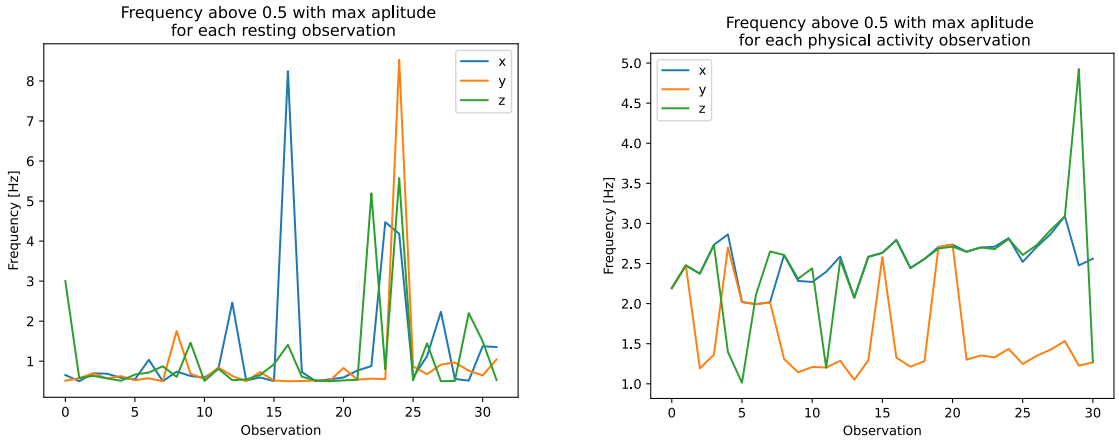
In our investigation of the frequency domain for the accelerometer data, we found that the x-accelerometer data has the most significant difference between rest and physical activity. Furthermore, we found that the frequencies with the highest amplitude for observation of rest and activity were around 0. However, when looking at fig. 6.3, which is an example of two observations, we see that for the physical activity observation, the frequency with the second highest amplitude is at around 2.73Hz, where for the resting observation there does not seem to exist another peak. This trend is seen throughout the remaining observations.



(a) Resting observation (b) Activity observation

Figure 6.3: Examples of observations after FFT in the x-direction from the WEEE dataset. The $x-$axis is the frequency, and the y-axis is the amplitude.

Thus, we have selected to use the frequency with the highest amplitude above 0.5Hz as our representation of the observation. Through fig. 6.4 we further validated that the data from the x-accelerometer signal has the most stable and distinct difference between resting and physical activity. Thus, we selected the FFT frequency threshold to classify physical activity when the highest amplitude above 0.5Hz (of the x-accelerometer signal) is between 2Hz to 3Hz. Using this range we obtain a confusion matrix seen in fig. 6.5, giving an accuracy of 0.96 and an F1-score of 0.89 for the combined WEEE and WESAD data.



(a) Resting observations. (b) Physical activity observations.

Figure 6.4: Max amplitude frequencies (above 0.5Hz) for resting and physical activity observations in the WEEE dataset. The x-axis is an observation, and the y-axis is the frequency.

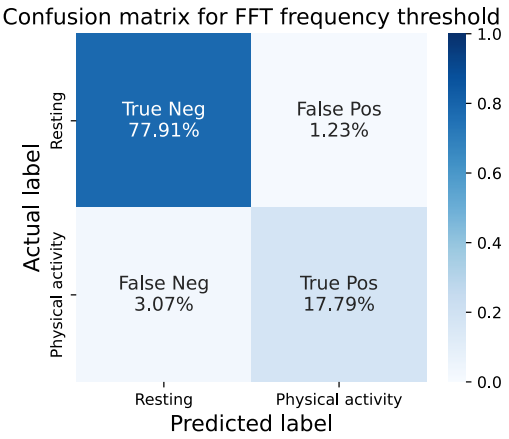


Figure 6.5: Confusion matrix of physical activity classification with frequency threshold between 2 and 3.

6.7.3 Confirming Threshold on Wrist Angel Activity Data

Before we selected the final activity classification model, we validated the models through data collected in the Wrist Angel study (a total of 23 observations). The data used for validation is recorded in a lab setting, however, the participants are the control group from the study, meaning the data comes from adolescents not diagnosed with OCD.

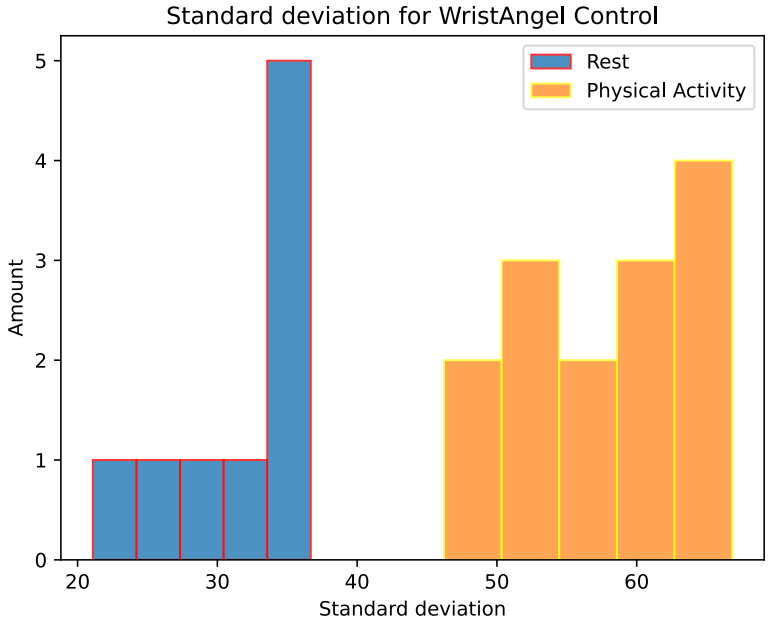


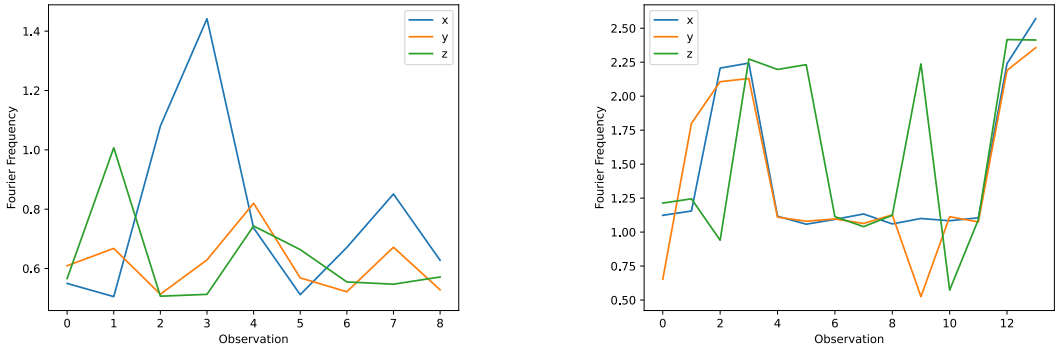
Figure 6.6: Histogram of standard deviations for resting and physical activity observations, respectively. From the Wrist Angel lab recordings. It is seen that a threshold of 40 makes a perfect split between resting and activity.

With the standard deviation threshold method, we have plotted a histogram shown in fig. 6.6. In the figure, we see that the standard deviation threshold of 40 is a perfect separation between resting and physical activity. Given that the maximum standard de-

viation of a resting observation is 36.7, and the minimum standard deviation of an activity observation is 46.2, this means that we have an accuracy and F1-score of 1.

On the other hand, our FFT frequency threshold method's performance is undesired. Through the plot of the frequency of the highest amplitude above 0.5, we see that the predefined range of 2 to 3 Hz does not seem to apply to the observations of physical activity. Additionally, it does not seem a clear new range can be defined through this method. With the predefined range, we obtain an accuracy of 0.57 and an F1-score of 0.44 for the validation set.

Because the standard deviation method worked better on the Wrist Angel lab recordings, we moved on with this method as our physical activity classifier for the Wrist Angel data.



(a) Resting observations.

(b) Physical activity observations.

Figure 6.7: Max amplitude frequencies (above 0.5Hz) for resting and physical activity observations from the Wrist Angel lab recordings. The x-axis is an observation, and the y-axis is the frequency.

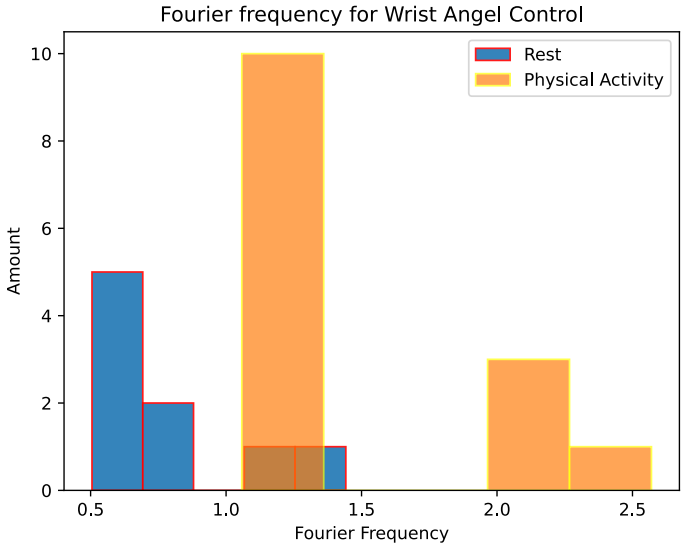


Figure 6.8: Histogram of the Fourier frequency in the x-direction for resting and physical activity observations respectively. From the Wrist Angel lab recordings. It is seen that there are no distinct good splits.

6.8 Results from Wrist Angel

In this section, we will first present a detailed evaluation of our model using a few of the defined scenarios on the Wrist Angel data. Thereafter, we will evaluate the general performance of our model based on all scenarios. Lastly, we will dive deeper into different analyses to obtain a better understanding of our model performance.

Each run in this section was done with the details listed in table 6.26.

	Details	Value
Patience	Total epochs with no improvement	30
Delta	Relative validation loss improvement **Model saved based on delta**	1%
Optimizer	Adam	Weight decay = $1\text{E}-04$
LR	Initial Learning Rate (LR)	Autoencoder = 0.1 Classification = 0.01
LR scheduler	Autoencoder: `ReduceLROnPlateau` Classification: `LambdaR`	`ReduceLROnPlateau = ('min', factor=0.75, threshold=Delta, threshold_mode='rel', patience=10)` `LambdaR = (gamma = 0.9)`
Loss Function	Autoencoder: `MSELoss` Classification: `BCEWithLogitsLoss`	Autoencoder: Loss calculated separately for each signal Classification: Positive weights balanced by $\text{neg}_c/\text{pos}_c$
Seed	Only for data processing	123
Batch Size	Only for training split	1-min: 320 5-min: 64

Table 6.26: Parameters used for all final runs on the Wrist Angel data. Notice model is saved based on delta, which is based on the validation loss

6.8.1 Evaluation of Test Split Generalizability

As specified in section 6.4 we will only use the metrics from the test data split for reporting the results from the Wrist Angel dataset, mainly as these evaluations are the best indication of the model's true performance. In this section, we will show the loss plot, accuracy, and F1-score metric, for three different scenarios as each of these scenarios has different train, validation, and test loss/metric characteristics. The following results are generally representative of the results in the other scenarios[3]

Baseline Scenario

The baseline scenario is with 1-min data, 0-min LPT, no activity filtering, and fine-tuning a pre-trained model with a random split.

	Train	Validation	Test
Acc	**0.45 (0.04)**	0.42 (0.05)	0.41 **(0.04)**
F1-score	**0.44 (0.02)**	0.41 **(0.02)**	0.43 **(0.02)**

Table 6.27: The reported numbers are average accuracy (Acc) and F1-score over 10 runs, with the standard deviation given in parentheses. They are of the baseline scenario. Bold numbers are the best performance for accuracy and F1-score across train, val, and test, and the bold numbers in the parentheses indicate the smallest standard deviation.

As is evident from table 6.27, the metrics for the test split are indicative of the other splits.

[3]Given the very large quantity of results (≈ 2000), no table or results will be provided in this report to support this assertion. If the reader is interested all results can be found in github.com/haraldsr/Thesis-on-OCD-prediction if you have been granted access.

A plot of losses from a random run can be seen in fig. 6.9. Loss figures are in PNG format not in SVG format due to the large number of loss figures, thus reducing storage needs.

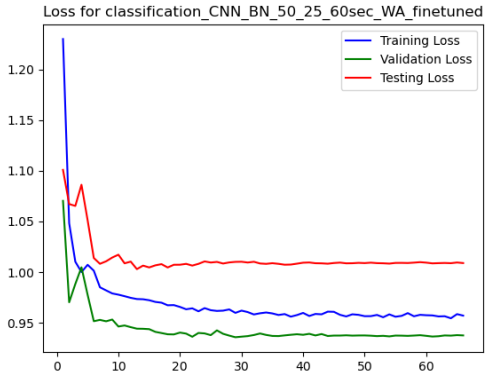


Figure 6.9: Plot of the classification fine-tuning loss from the scenario in table 6.27 for a randomly chosen run. The $x-$axis is epochs, and the $y-$axis is the binary cross-entropy loss.

From fig. 6.9 it can be seen that the model learns as expected, but it flattens out quickly.

Pre-Trained Personalized Scenario
However, for the personal fine-tuning scenarios, the loss curve for the classification looks different as seen in fig. 6.10.

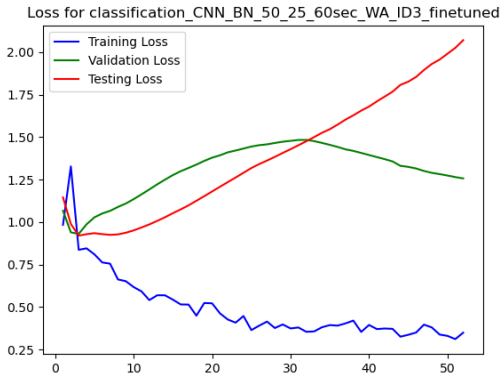


Figure 6.10: A plot of the classification fine-tuning loss from the personalized split on 1-minute data, 0-minute lead prediction time without activity classification. The loss curve is for the first run for ID 3. The $x-$axis is epochs, and the $y-$axis is the binary cross-entropy loss.

Figure 6.10 is the general trend for the personalized fine-tuning scenarios. It is seen that the model overfits. It learns to classify the training data, and sometimes the validation data, but does not generalize to the test data.

	Train	Validation	Test
Acc	0.59 (0.00)	**0.73** (0.08)	0.65 (0.08)
F1-score	0.43 (0.10)	**0.45** (0.07)	0.30 (0.05)

Table 6.28: The reported numbers are average accuracy (Acc) and F1-score over 10 runs, with the standard deviation given in parentheses. They are of the scenario with a fine-tuned pre-trained model on a personalized split with ID 3 as the test person. With 1-minute data, 0-minute lead prediction time, and without the activity filter. Bold numbers are the best performance for accuracy and F1-score across train, val, and test, and the bold numbers in the parentheses indicate the smallest standard deviation.

In table 6.28 the Accuracy and F1-score metrics are given, and although the losses are different, the scores for the different splits are similar. Training and validation both have better scores, but this is as expected, given the model is trained on training data, and saved based on the validation data as explained in table 6.26.

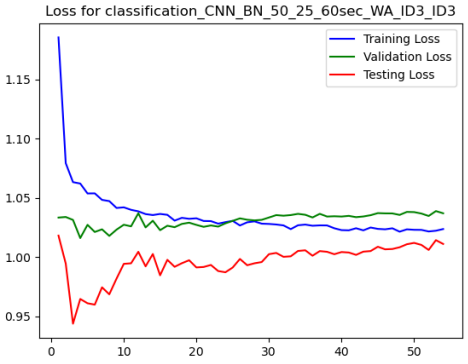
Train Personalized Scenario

Lastly, for the direct personal training on the Wrist Angel dataset, we do not see a general trend, but different loss curves based on the participant, see fig. 6.11.

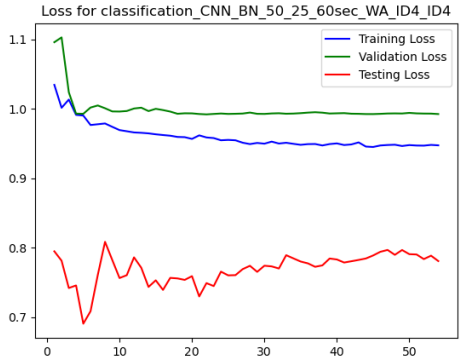
Firstly from fig. 6.11 it can be seen that IDs 2 and 3 are missing, they were excluded due to them not having generated enough data. These IDs are excluded throughout the results when looking at personalized splits. Additionally, from fig. 6.11 it is clear that for no participant, the model is learning well nor is generalizable, but again, given our model save condition as explained in table 6.26 only the point of minimal loss is indicative of final model performance. From table 6.29 it can be seen that even though the performance of the train split is better than the test split, it is not great. Thus, the conclusions from the test metrics are still indicative of the overall model performance.

	Train		Validation		Test	
	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
ID 3	**0.40** (0.06)	**0.44** (0.03)	0.22 (0.16)	0.30 (0.14)	0.31 (0.09)	0.32 (0.02)
ID 4	0.44 (0.21)	**0.41** (0.06)	0.25 (0.35)	0.00 (0.00)	**0.59** (0.21)	0.18 (0.05)
ID 5	**0.52** (0.19)	**0.46** (0.09)	0.36 (0.04)	0.30 (0.04)	0.40 (0.08)	0.26 (0.01)
ID 6	0.65 (0.01)	**0.10** (0.14)	**0.76** (0.00)	0.00 (0.00)	0.70 (0.01)	0.05 (0.02)
ID 7	0.66 (0.00)	0.00 (0.00)	0.70 (0.00)	0.00 (0.00)	0.67 (0.00)	0.00 (0.00)
ID 8	0.64 (0.00)	0.01 (0.02)	0.74 (0.00)	0.00 (0.00)	0.68 (0.01)	0.06 (0.01)

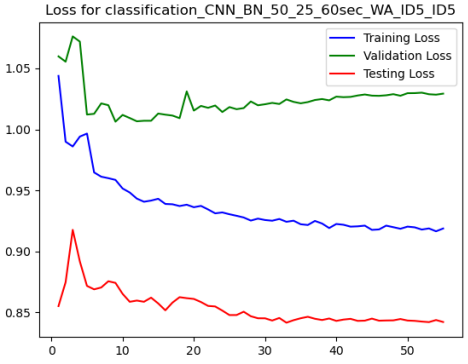
Table 6.29: Average accuracy and F1-score for each participant for the personalized direct train scenario. With 1-minute data, 0-minute LPT, and no activity filter. Standard deviation reported in parenthesis. Bold numbers are the best performance for each participant across train, validation, and test, and the bold numbers in the parentheses indicate the smallest standard deviation. The bottom two rows are not bolded as the predictions are biased towards the majority class leading to an F1-score of (almost) zero.



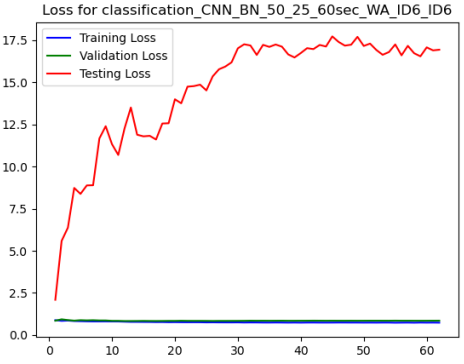
(a) ID 3



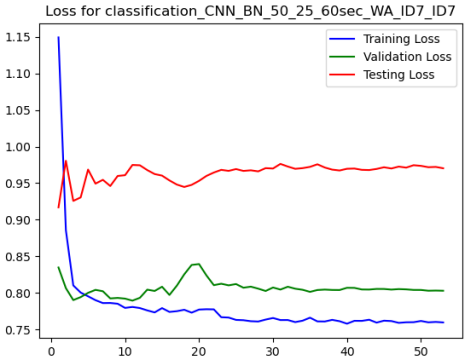
(b) ID 4



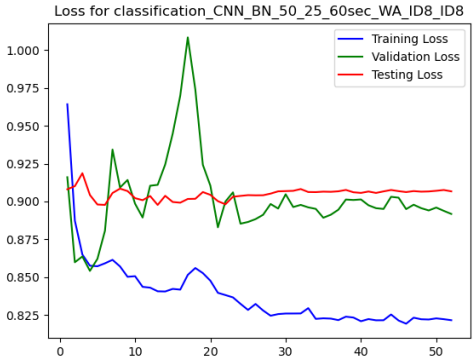
(c) ID 5



(d) ID 6



(e) ID 7



(f) ID 8

Figure 6.11: Loss curves for the direct personalized training on Wrist Angel with each participant as test person. With the 1-min data, 0-minute LPT, without activity filter. The $x-$axis is epochs, and the $y-$axis is the binary cross-entropy loss.

6.8.2 All Wrist Angel Scenarios

Below in tables 6.30 to 6.33 the evaluations of the model on the test split of the Wrist Angel data are presented for all 120 scenarios. The tables are divided into four tables where two tables are for the 5-minute interval, one with the activity filter (table 6.30), and one without (table 6.31), and the two tables for the 1-minute interval, one with the activity filter (table 6.32), and one without (table 6.33). Each of these tables presents evaluations for the different applications of the model and for the different lead prediction times.

		Lead prediction time						
		0-min	1-min	2-min	3-min	4-min	5-min	$\Sigma\mu$
Predict	Acc	0.42 (0.01)	0.48 (0.02)	0.47 (0.01)	0.42 (0.01)	0.45 (0.01)	0.46 (0.01)	0.40
	F1-score	0.33 (0.01)	0.34 (0.01)	0.44 (0.01)	0.33 (0.01)	0.34 (0.01)	0.37 (0.01)	
Pre-trained random	Acc	0.50 (0.14)	0.48 (0.15)	0.53 (0.03)	0.40 (0.05)	0.45 (0.08)	0.44 (0.03)	**0.43**
	F1-score	0.31 (0.13)	0.30 (0.08)	0.41 (0.08)	**0.45** (0.01)	0.42 (0.07)	**0.45** (0.02)	
Pre-trained personalized	Acc	0.41 (0.23)	**0.56** (0.16)	0.40 (0.15)	0.47 (0.18)	0.43 (0.15)	0.37 (0.14)	0.36
	F1-score	0.25 (0.18)	0.28 (0.19)	0.27 (0.19)	0.27 (0.16)	0.29 (0.14)	0.34 (0.14)	
Train random	Acc	0.42 (0.12)	0.52 (0.10)	**0.56** (0.07)	0.41 (0.11)	0.55 (0.10)	0.44 (0.11)	0.41
	F1-score	0.38 (0.07)	0.25 (0.12)	0.27 (0.18)	0.42 (0.07)	0.31 (0.18)	0.41 (0.09)	
Train personalized	Acc	0.41 (0.27)	0.43 (0.25)	0.36 (0.21)	0.39 (0.22)	0.40 (0.17)	0.28 (0.18)	0.31
	F1-score	0.16 (0.12)	0.20 (0.16)	0.27 (0.18)	0.18 (0.14)	0.26 (0.15)	0.34 (0.12)	
Mean		0.36	0.37	**0.40**	0.38	**0.40**	0.39	0.38

Table 6.30: Results for 5-min window length, with physical activity filter. The reported numbers are average accuracy (Acc) and F1-score over 10 runs, with the standard deviation given in parentheses. The scenario with the best sum of evaluation (Acc + F1) is highlighted in green. Bold numbers are the best performance for the overall accuracy and F1-score, and the bold numbers in the parentheses indicate the smallest standard deviation. Furthermore, the highest means are bolded, which are the last row and the rightmost column.

		Lead prediction time						
		0-min	1-min	2-min	3-min	4-min	5-min	$\Sigma\mu$
Predict	Acc	0.42 (0.01)	0.48 (0.01)	0.46 (0.01)	0.41 (0.01)	0.45 (0.01)	0.46 (0.01)	0.40
	F1-score	0.33 (0.01)	0.35 (0.01)	0.43 (0.01)	0.33 (0.01)	0.35 (0.01)	0.37 (0.01)	
Pre-trained random.	Acc	0.42 (0.07)	0.38 (0.14)	**0.53** (0.04)	0.45 (0.11)	0.46 (0.07)	0.35 (0.05)	0.41
	F1-score	0.35 (0.06)	0.36 (0.11)	0.42 (0.14)	0.38 (0.15)	0.43 (0.04)	**0.44** (0.02)	
Pre-trained personalized	Acc	0.40 (0.22)	0.46 (0.20)	0.42 (0.20)	0.50 (0.12)	0.43 (0.15)	0.41 (0.14)	0.37
	F1-score	0.24 (0.16)	0.27 (0.14)	0.29 (0.18)	0.26 (0.18)	0.32 (0.15)	0.41 (0.17)	
Train random	Acc	0.46 (0.10)	0.45 (0.10)	0.52 (0.08)	0.48 (0.11)	0.46 (0.09)	0.39 (0.07)	**0.42**
	F1-score	0.31 (0.08)	0.35 (0.08)	0.39 (0.17)	0.40 (0.09)	0.42 (0.07)	0.43 (0.05)	
Train personalized	Acc	0.45 (0.23)	0.42 (0.25)	0.36 (0.23)	0.36 (0.21)	0.39 (0.18)	0.38 (0.18)	0.31
	F1-score	0.18 (0.13)	0.21 (0.15)	0.21 (0.15)	0.23 (0.16)	0.26 (0.15)	0.26 (0.14)	
Mean		0.36	0.37	**0.40**	0.38	**0.40**	0.39	0.38

Table 6.31: Results for 5-min window length, without physical activity filter. The reported numbers are the average accuracy (Acc) and F1-score over 10 runs, with the standard deviation given in parentheses. The scenario with the best sum of evaluation (Acc + F1) is highlighted in green. Bold numbers are the best performance for the overall accuracy and F1-score, and the bold numbers in the parentheses indicate the smallest standard deviation. Furthermore, the highest means are bolded, which are the last row and the rightmost column.

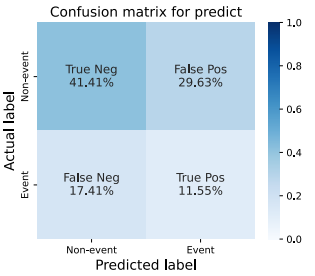
		Lead prediction time						
		0-min	1-min	2-min	3-min	4-min	5-min	$\Sigma\mu$
Predict	Acc	0.55 (0.01)	0.56 (0.01)	0.51 (0.01)	0.49 (0.01)	0.46 (0.01)	0.49 (0.02)	0.42
	F1-score	0.33 (0.01)	0.36 (0.01)	0.34 (0.01)	0.33 (0.01)	0.27 (0.02)	0.34 (0.02)	
Pre-trained random	Acc	0.45 (0.03)	0.53 (0.11)	0.41 (0.04)	0.45 (0.02)	0.41 (0.03)	0.46 (0.02)	**0.44**
	F1-score	0.39 (0.03)	0.29 (0.09)	0.46 (0.01)	0.49 (0.01)	0.49 (0.02)	**0.50** (0.02)	
Pre-trained personalized	Acc	**0.64** (0.14)	0.59 (0.11)	0.44 (0.16)	0.60 (0.17)	0.46 (0.08)	0.57 (0.15)	0.41
	F1-score	0.22 (0.14)	0.25 (0.12)	0.23 (0.10)	0.26 (0.20)	0.33 (0.11)	0.31 (0.21)	
Train random	Acc	0.43 (0.08)	0.45 (0.09)	0.46 (0.04)	0.45 (0.03)	0.47 (0.07)	0.42 (0.05)	**0.44**
	F1-score	0.40 (0.03)	0.36 (0.09)	0.47 (0.02)	0.48 (0.03)	0.41 (0.11)	0.49 (0.02)	
Train personalized	Acc	0.54 (0.22)	0.51 (0.18)	0.46 (0.22)	0.61 (0.21)	0.36 (0.20)	0.39 (0.17)	0.33
	F1-score	0.13 (0.13)	0.15 (0.10)	0.13 (0.13)	0.16 (0.13)	0.24 (0.18)	0.26 (0.14)	
Mean		0.41	0.40	0.40	**0.43**	0.39	0.42	0.41

Table 6.32: Results for 1-min window length, with physical activity filter. The reported numbers are the average accuracy (Acc) and F1-score over 10 runs, with the standard deviation given in parentheses. The scenario with the best sum of evaluation (Acc + F1) is highlighted in green. Bold numbers are the best performance for the overall accuracy and F1-score, and the bold numbers in the parentheses indicate the smallest standard deviation. Furthermore, the highest means are bolded, which are the last row and the rightmost column.

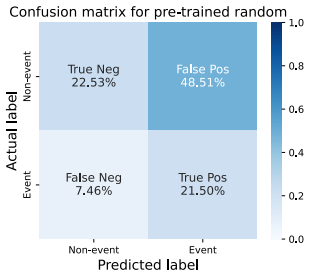
		Lead prediction time						
		0-min	1-min	2-min	3-min	4-min	5-min	$\Sigma\mu$
Predict	Acc	0.54 (0.01)	0.55 (0.01)	0.50 (0.01)	0.49 (0.01)	0.46 (0.02)	0.49 (0.02)	0.41
	F1-score	0.33 (0.02)	0.36 (0.02)	0.33 (0.01)	0.33 (0.02)	0.26 (0.02)	0.32 (0.02)	
Pre-trained random	Acc	0.41 (0.04)	0.50 (0.10)	0.43 (0.03)	0.43 (0.05)	0.53 (0.02)	0.47 (0.05)	**0.45**
	F1-score	0.43 (0.02)	0.32 (0.11)	0.46 (0.01)	**0.50** (0.01)	0.43 (0.09)	0.48 (0.03)	
Pre-trained personalized	Acc	0.60 (0.17)	0.54 (0.12)	0.48 (0.11)	**0.62** (0.14)	0.48 (0.10)	0.54 (0.14)	0.41
	F1-score	0.23 (0.12)	0.27 (0.14)	0.26 (0.10)	0.26 (0.17)	0.29 (0.12)	0.34 (0.18)	
Train random	Acc	0.47 (0.11)	0.54 (0.10)	0.42 (0.07)	0.43 (0.05)	0.51 (0.04)	0.49 (0.09)	**0.45**
	F1-score	0.37 (0.12)	0.28 (0.13)	0.46 (0.03)	0.46 (0.06)	0.46 (0.02)	0.44 (0.12)	
Train personalized	Acc	0.56 (0.17)	0.51 (0.20)	0.46 (0.26)	0.51 (0.18)	0.37 (0.21)	0.41 (0.17)	0.32
	F1-score	0.14 (0.12)	0.13 (0.12)	0.13 (0.14)	0.20 (0.15)	0.22 (0.18)	0.22 (0.17)	
Mean		0.41	0.40	0.39	0.42	0.40	**0.42**	0.41

Table 6.33: Results for 1-min window length, without physical activity filter. The reported numbers are the average accuracy (Acc) and F1-score over 10 runs, with the standard deviation given in parentheses. The scenario with the best sum of evaluation (Acc + F1) is highlighted in green. Bold numbers are the best performance for the overall accuracy and F1-score, and the bold numbers in the parentheses indicate the smallest standard deviation. Furthermore, the highest means are bolded, which are the last row and the rightmost column.

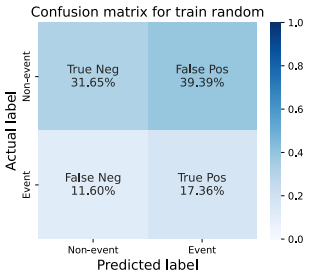
To further investigate the performance of these application methods, we will examine 5 scenarios, specifically with 1-minute intervals, 0-minute lead prediction time, and no activity classification, that is, scenarios present in table 6.33.



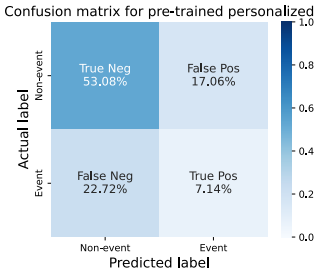
(a) Confusion matrix for predict.



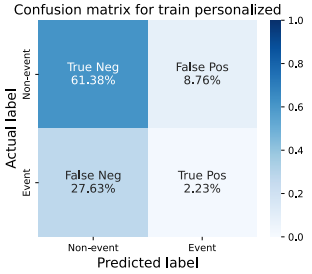
(b) Confusion matrix for pre-trained random.



(c) Confusion matrix for train random.



(d) Confusion matrix for pre-trained personalized.



(e) Confusion matrix for train personalized.

Figure 6.12: Confusion matrix for the five different scenarios with a 1-minute window length, without activity classification and no lead prediction time. For the prediction and the two scenarios with random split, we have averaged across all 10 runs. For scenarios with personalized split, we have averaged across participants over all runs. All scenarios use all 3 data splits: train, validation, and test sets. A participant-based confusion matrix for pre-trained personalized can be found in fig. A.12.

The performances of the five scenarios are presented in fig. 6.12. We are using the predicted scenario as a baseline for our comparison. For scenarios with random split, we see a shift from predicting observations as true negative to false positive when the WA dataset is used to train the model. However, this also increases true positives and lowers false negatives, because the model is more willing to predict observations as an event. On the contrary, for personalized split scenarios, we see a minor change in the opposite direction, where generally speaking more observations are predicted as non-events. However, differences between participants are observed, such as ID5 in pre-trained personalized has a significantly different distribution than the other participants (fig. A.12). The same trend for random split scenarios is not seen in the other datasets shown in fig. A.11.

6.9 Post Hoc Analysis

In this section the model performance results presented in section 6.8 are analyzed. First, we analyze the effect of the factors tested: lead prediction time, activity, window length, and application of the model. Hereafter, the difference in performance between the participants for personalized splits is examined. Finally, we have performed a threshold analysis to investigate whether this could increase the model performance.

The factors' effects have been tested with Mann-Whitney U rank tests. All computed p-values presented in this section have been adjusted according to the Benjamini-Hochberg correction (see section 5.4.3) with 54 tests. 2 tests for activity (1 accuracy and 1 F1-score), 2 tests for window length (1 accuracy and 1 F1-score), 30 tests for lead prediction time (15 accuracy and 15 F1-score), and 20 tests for application of model (10 accuracy and 10 F1-score). In total: $2 + 2 + 30 + 20 = 54$. In table 6.37 an overview of a few selected Mann-Whitney U rank tests is presented. These tests are commented on in the following sections.

		Accuracy			F1-score		
Baseline model	Test variable	Baseline μ	Test variable μ	p-value	Baseline μ	Test variable μ	p-value
1-minute data 0-minute lead Pre-trained random No activity	Including activity	0.41	0.45	**0.011**	0.43	0.39	**0.003**
1-minute data 0-minute lead Pre-trained random No activity	5-minute data	0.41	0.42	0.954	0.43	0.35	**0.002**
1-minute data 0-minute lead Pre-trained random No activity	Train random	0.41	0.47	0.569	0.43	0.37	0.330
1-minute data 0-minute lead Pre-trained random No activity	Pre-trained personalized	0.41	0.60	**0.017**	0.43	0.23	**9E-04**

Table 6.34: Overview of selected Mann-Whitney U rank tests. P-values are adjusted according to the Benjamini-Hochberg correction with 54 tests. Bold p-values are significant on a 5% significance level.

6.9.1 Lead Prediction Time (LPT)

By comparing the lead prediction times for each window length and with or without physical activity (bottom row in each table, tables 6.30 to 6.33) we see that having a lead prediction time over 1 minute performs better. However, it is only a marginal improvement as it is below 5 percentage points of difference. Therefore, we are interested in testing if there is a significant difference between the lead prediction using the Mann-Whitney U rank test. We are testing for significant differences between the lead prediction time using only one scenario, pre-trained random with the 1-minute dataset without physical classification.

	Accuracy						F1-score					
Lead prediction time	0	1	2	3	4	5	0	1	2	3	4	5
0	-	0.052	0.108	0.162	**0.001**	**0.007**	-	**0.005**	**0.002**	**9E-04**	0.076	**0.009**
1	0.052	-	0.292	0.104	0.237	0.954	**0.005**	-	**0.001**	**9E-04**	**0.007**	**0.001**
2	0.108	0.292	-	0.349	**0.001**	0.138	**0.002**	**0.001**	-	**9E-04**	0.740	0.354
3	0.162	0.104	0.349	-	**0.005**	**0.015**	**9E-04**	**9E-04**	**9E-04**	-	**0.003**	0.076
4	**0.001**	0.237	**0.001**	**0.005**	-	**0.002**	0.076	**0.007**	0.740	**0.003**	-	0.281
5	**0.007**	0.954	0.138	**0.015**	**0.002**	-	**0.009**	**0.001**	0.354	0.076	0.281	-

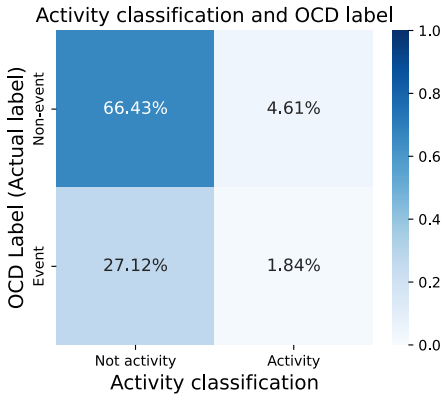
Table 6.35: P-values for Mann-Whitney U rank test between each lead prediction time for the pre-trained random application on the 1-minute dataset with no activity classification. P-values are adjusted according to the Benjamini-Hochberg correction with 54 tests. The bold p-values are significant on a 5% significance level.

From tables 6.33 and 6.35 we see the four-minute lead prediction time having a significantly higher accuracy (0.53) than all other LPTs, except the one-minute LPT (which has an accuracy of 0.50). However, this is not the case for the F1-score. Here the one-minute LPT is notable as it has a significantly lower F1-score (0.32) than all other LPTs. Furthermore, the three-minute LPT has the highest F1-score (0.50), which is significantly higher than all but the five-minute LPT (which has an F1-score of 0.48). In general, it is not possible to determine the best LPT, due to varying results for accuracy and F1-score. However, there is no evidence that a zero-minute LPT is significantly better than having a non-zero LPT. On the contrary, it seems increasing the LPT affects the model positively.

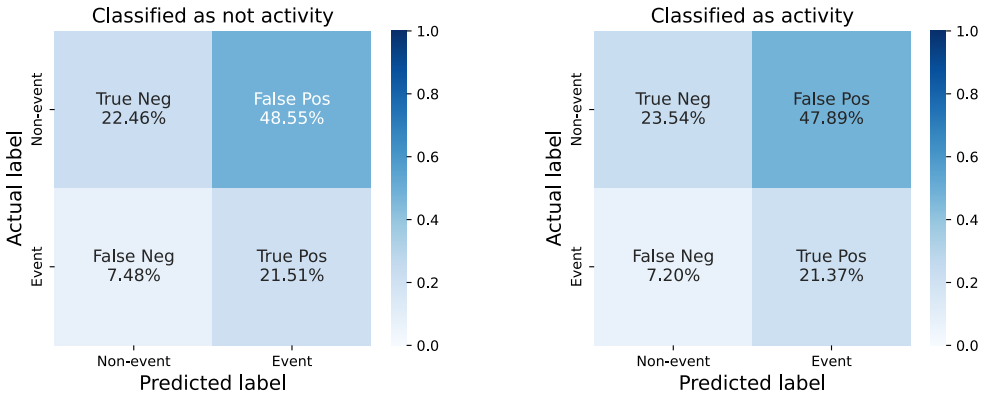
6.9.2 Activity Model

Comparing performance with and without physical activity (bottom right corner between tables 6.30 and 6.31, and tables 6.32 and 6.33) we see no difference in performance. Furthermore, we specifically analyze the 1-min data pre-trained random scenario and test for significance between including activity classification or not, with the Mann-Whitney U rank test, for 0-min lead prediction time. The test results can also be seen in table 6.34. After adjusting the p-values with the Benjamini-Hochberg correction we get a $p - value = 0.011$ for accuracy and $p - value = 0.003$ for F1-score. Hence the difference between with and without activity classification in this situation is significant, with a 5% significance level, but opposite. With activity, the accuracy of 0.45 is significantly higher than without, where the accuracy is 0.41. In contrast to the F1-score, which is 0.43 without activity. This is significantly higher than the F1-score of 0.39 when activity classification is included. Generally, based on the contradictory results and the similar performance, there is no support for including activity classification would improve the performance.

In addition, we investigated the observation distribution after our classification to determine if a trend can be seen.



(a) Visualization of classification model and the true OCD label.



(b) Confusion matrix for observations not classified as activity.

(c) Confusion matrix of observations classified as activity.

Figure 6.13: Confusion matrix for pre-train random using 0-minute lead prediction time with 1-minute window length. Average across 10 runs. In fig. 6.13b we have observations that are not classified as activity. In fig. 6.13c are observations classified as activity.

In fig. 6.13 we evaluate the performance of the activity classification. In fig. 6.13a we see that 94% of the observations were classified as not in activity. The activity classification had the purpose of reducing the false positive rate based on the assumption that an OCD event and physical activity have similar physiological data such as increased HR described in section 4.2.2. Thus the desired outcome is to minimize true OCD events, which are classified as activity. In figs. 6.13b and 6.13c, through the distribution expressed in colors, we see that, generally, the model suffers from false positives; this is present for both observations classified as activity and observations not classified as activity. Ideally, for activity classification to work well, there should be few false positives in fig. 6.13b, but many in fig. 6.13c, because this would mean that the activity filter removes many false positives.

Furthermore, we examine if there are differences in the activity classification between participants. In table 6.36 we see that the average percentage of observations classified as activity does not differ between the participants.

	Percentage activity	Percentage with activity and OCD event (actual label)
ID3	4.4%	0.5%
ID4	4.7%	1.4%
ID5	6.2%	0.5%
ID6	6.4%	2.1%
ID7	6.1%	2.1%
ID8	6.5%	2.1%

Table 6.36: The distribution of activity classification for each participant used for the pre-trained personalized model. Using 1-minute data, 0-minute LPT.

6.9.3 Window Length

Comparing the performance of the window lengths, 1- and 5-min in the bottom right corner between tables 6.30 and 6.32, and tables 6.31 and 6.33. Here, it is seen that the 1-minute window length performs better, although the performance improvement is only 3 percentage points. This is supported by our test, which focuses on the pre-trained random model without activity classification and a lead prediction time of zero minutes. We use the Mann-Whitney U rank test to see if there is a significant difference in accuracy and F1-score between using a window length of one minute vs a window length of five minutes. The test results can also be seen in table 6.34. Here, the average accuracy of the 1-minute data is 0.41, and for the 5-minute data, it is 0.42. After correcting the p-values with the Benjamini-Hochberg method the test gives a $p - value = 0.954$, hence there is no significant difference in accuracy between 1-minute window length and 5-minute window length. For the F1-score, the p-value is 0.002. Therefore, the use of 1-minute data has a significantly higher F1-score of 0.43, than the use of 5-minute data, where the F1-score is 0.35.

Generally, we can say that there is no indication that it should be advantageous to include five minutes of data in each observation instead of one minute.

6.9.4 Application of Model

Looking at tables 6.30 to 6.33 we see that applications with a random split (pre-trained random and train random) on average perform the best. Then comes the direct prediction and the pre-trained personalized application. The worst is the personalized train, without a pre-trained model.

To compare the different applications of the model, we executed Mann-Whitney U tests between applications, on the 1-minute data, with 0-minute lead prediction time and without the activity filter. The Benjamini-Hochberg corrected p-values of these tests can be seen in table 6.37.

	Accuracy					F1-score				
Model application type	Predict	PT random	PT personal	T random	T personal	Predict	PT random	PT personal	T random	T personal
Predict	-	**0.001**	**0.024**	0.138	0.570	-	**9E-04**	**0.016**	**0.045**	**0.001**
PT random	**0.001**	-	**0.017**	0.569	0.104	**9E-04**	-	**9E-04**	0.330	**9E-04**
PT personal	**0.024**	**0.017**	-	0.076	0.954	**0.016**	**9E-04**	-	**0.005**	0.125
T random	0.138	0.569	0.076	-	0.313	**0.045**	0.330	**0.005**	-	**0.003**
T personal	0.570	0.104	0.954	0.313	-	**0.001**	**9E-04**	0.125	**0.003**	-

Table 6.37: P-values for Mann-Whitney U rank test between each application type of the model for the 1-minute dataset with no activity classification and 0-minute lead prediction time. P-values are adjusted according to the Benjamini-Hochberg correction with 54 tests. The bold p-values are significant on a 5% significance level.

Firstly, from tables 6.33 and 6.37 it can be seen that the pre-trained random application has the lowest accuracy of 0.41, which is significantly lower than the direct prediction scenario (0.54) and the pre-trained personalized applications. Additionally, the pre-trained personal application has the highest accuracy of 0.60, significantly better than predicting directly and the pre-trained random. On the other hand for the F1-score, more significant differences are seen. The pre-trained model with random split fine-tuning has the best F1-score of 0.43, and this is significantly better than all applications except the training from scratch on the Wrist Angel data with a random split. Furthermore, the personalized training on Wrist Angel data has the lowest F1-score of 0.14. This is significantly lower than all the other application types, except using the pre-trained model, with personalized fine-tuning. Hence, overall, it seems the random split applications perform the best, whether using a pre-trained model or not. Additionally, the personalized applications perform the worst regarding F1-score, but good regarding accuracy. In most cases, this is due to the model mostly predicting the majority class, namely non-event. The high standard deviation for the personalized models should also be taken into account, as it indicates large performance variations between participants. This is further investigated in section 6.9.5.

Pre-Trained

To examine whether the use of a pre-trained model would improve performance, we can look at the sum of the means column in each table (tables 6.30 to 6.33) and compare rows two (pre-trained random), four (training random), three (pre-trained personalized), and five (training personalized). First, comparing pre-trained random with train random, we see no significant improvement, as the pre-trained model performs best with activity classification, but performs worse without for the 5-min window. For the 1-minute window, the models perform the same. However, personalized models improve performance by an average of seven percentage points when the pre-trained model is used. This indicates that using a pre-trained model does not worsen performance.

Focusing on 1-minute data, without physical activity classification, and 0-minute LPT, we conduct Mann-Whitney U rank tests for, respectively, the random and the personalized scenarios between using a pre-trained model or training directly on the Wrist Angel data, see table 6.37. For the random scenario, we get $p-value = 0.569$, and $p-value = 0.330$ for the accuracy and F1-score, respectively. Hence we cannot reject the null hypothesis, and there is no significant difference between using a pre-trained model or not in this case. For the personalized case, we see the same thing. The p-values for accuracy and F1-score are 0.954 and 0.125, therefore, there is no significant difference in using a pre-trained model. Overall it seems the pre-trained model, does not affect the performance significantly.

6.9.5 Results by Participant

To analyze model performance for personalized fine-tuning based on the different participants, we have plotted the difference in model performance by ID (fig. 6.14).

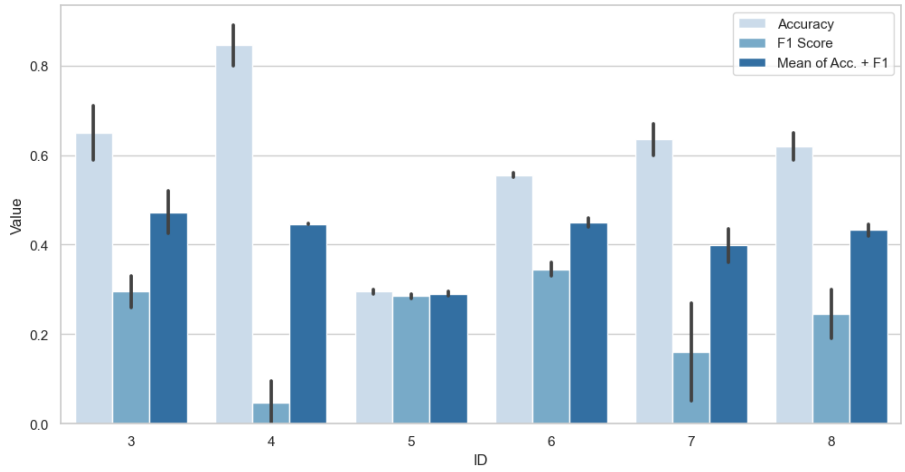


Figure 6.14: Accuracy and F1-score for fine-tuning the pre-trained model to the different participants. Averaged over 2 runs, with the standard deviation shown as error bars. The data used is from the baseline scenario.

From fig. 6.14 we see differences in performance between the participants. For ID 4 we see it's heavily biased toward predicting as non-event (see fig. A.12b) given the high accuracy. For IDs 3, 6, and 8, the accuracy is lower, but the F1-score is higher, thus giving a more balanced performance, with ID 3 having the highest average accuracy and F1-score. Finally, for ID 5 the accuracy and F1-score are equal, however, the accuracy is 0.3[4].

As can be seen from table 6.33 and section 6.9.1 a non-zero lead prediction time performs better. This could perhaps be explained by some participants tagging OCD events at the end of the event instead of at the beginning. To investigate this claim fig. 6.15 is a plot of performance by ID and lead prediction time.

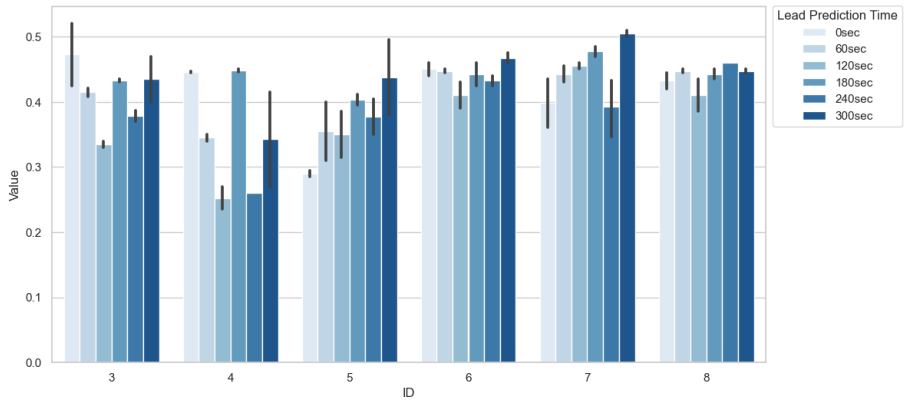


Figure 6.15: Average accuracy and F1-score for the different participants and lead prediction times. Averaged over 2 runs, with the standard deviation shown as error bars. The data used is 1-minute data and no activity filter.

[4]Negating the prediction would yield an accuracy of 0.7, however, the resulting F1-score would be 0.1

From fig. 6.15 the superior non-zero lead prediction time performance does not seem to be true between participants. For ID 3 a 0 LPT performs best, but for ID 7 a 5-min LPT performs best, ie the opposite being true. For IDs 6 and 8, the performance is approximately uniform across LPT. For IDs 5 and 7, performance appears to be linearly proportional to LPT. Finally, for ID 4 we see a varying performance across LPTs perhaps hinting at inconsistent tagging.

For both figs. 6.14 and 6.15 each result for a given ID was only run twice. The (black) error bars are the standard deviation between results for each run, and from the plots, it can be seen that the deviation between runs is low. However, each interpretation should be taken with this uncertainty in mind.

6.9.6 Threshold Analysis

It is possible to improve the performance by looking at the threshold of when an event is predicted as an OCD event. The optimal threshold has been found in 4 ways: with the ROC curve and by maximizing both accuracy and/or F1-score. The model used in the comparison in table table 6.38 is the baseline scenario with 1-min and 5-min window length (column 1 and row 2 in tables 6.31 and 6.33).

		Threshold	Accuracy	F1-score	μ
1-min	Baseline	0.5	0.41 (0.04)	0.43 (0.02)	0.42
	ROC Curve	0.5 (0.01)	0.54 (0.13)	0.37 (0.09)	**0.45**
	Accuracy	0.79 (0.12)	**0.69 (0.00)**	0.02 (0.02)	0.35
	F1-score	0.43 (0.04)	0.33 (0.01)	**0.48 (0.00)**	0.41
	Accuracy + F1-score	0.52 (0.01)	0.57 (0.06)	0.34 (0.04)	**0.45**
5-min	Baseline	0.5	0.42 (0.07)	0.35 (0.06)	0.38
	ROC Curve	0.49 (0.01)	0.64 (0.16)	0.1 (0.17)	0.37
	Accuracy	0.96 (0.03)	**0.72 (0.00)**	0.01 (0.01)	0.36
	F1-score	0.41 (0.10)	0.28 **(0.00)**	**0.48 (0.00)**	0.36
	Accuracy + F1-score	0.53 (0.02)	0.57 (0.02)	0.23 (0.02)	**0.40**

Table 6.38: Results from finding the optimal threshold with ROC curve and maximizing for accuracy and F1-score. The reported numbers are the average threshold, accuracy, and F1-score over 10 runs, with the standard deviation given in parentheses. The last column, μ, is the average of accuracy and F1-score. Bold numbers are the best performance for each window length across all methods, and the bold numbers in the parentheses indicate the smallest standard deviation.

From table 6.38 it is clear that a significant overall improvement in performance cannot be found by optimizing the threshold. Optimizing for the accuracy or the F1-score will lead to a significant improvement for the metric, but will come at the cost of the other. In fig. 6.16 a random run for optimization is presented.

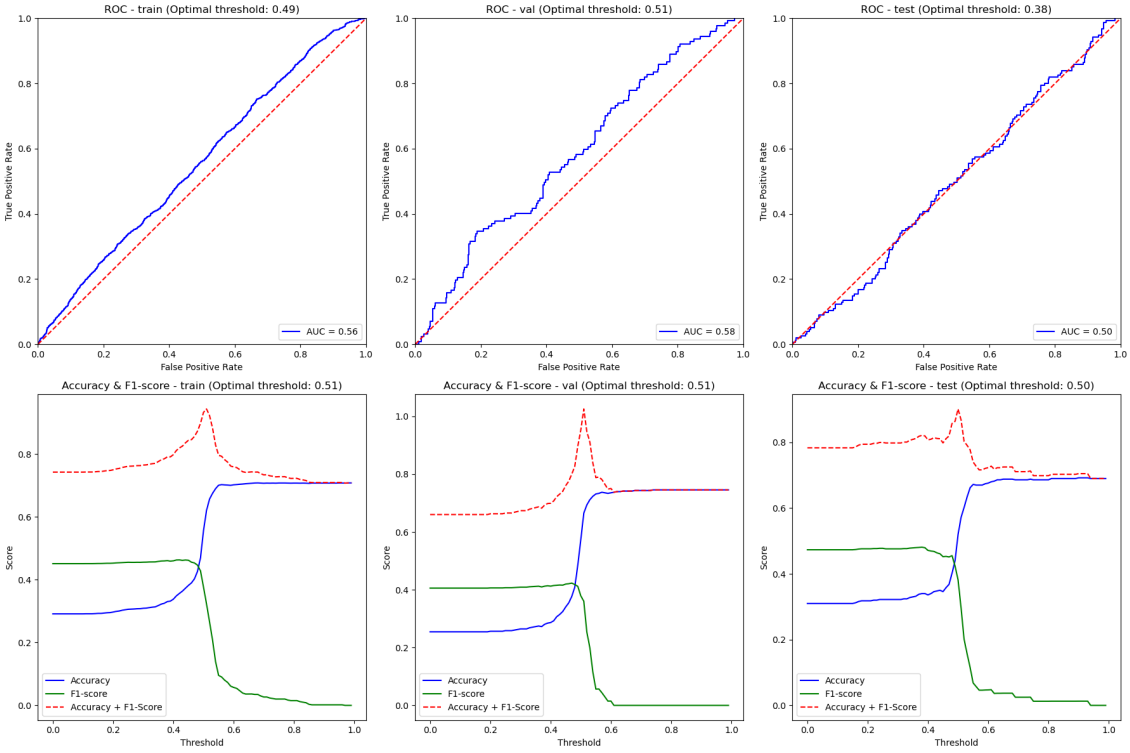


Figure 6.16: The top 3 plots are ROC curves from training, validation, and test data. The bottom 3 plots are accuracy and F1-score by threshold on the $x-$axis, as well for training, validation, and test data. The blue line is accuracy, the green is F1-score, and the dashed red is accuracy + F1-score. The optimal threshold for accuracy and F1-score is found by maximizing according to accuracy + F1-score.

From fig. 6.16 it can be seen that the ROC curve is close to linear and AUC = 0.5, thus not better than random guessing. Additionally, it can be seen that accuracy and F1-score are reversely proportional, whereby increasing the threshold will reduce the F1-score but increase accuracy.

7 Discussion

In the discussion, we have focused more on discussing our model and the reasons for the poor performance on the Wrist Angel dataset. First, we will discuss the differences in data between the open-source and the WA datasets. This is followed by discussions of the model choices and mixed results of transfer learning. Then the different factors outlined in this thesis, like the window length, are considered. Given the lackluster performance, we will also discuss our model application strategy and whether the greater emphasis should have been on the WA dataset, like through self-supervised learning. Finally, we will discuss the balance between overpredicting or underpredicting OCD events and how our model compares with the Wrist Angel feasibility study by [Lønfeldt et al. 2023].

7.1 Data

We knew we would face a challenge since all open-source datasets we used consisted of data from adults in controlled settings, except for the ADARP dataset which was collected in the wild. This is in contrast to the Wrist Angle dataset, which was collected from adolescents in the wild. The physiological signals have different properties as mentioned in section 4.2. However, when we looked at the differences between the datasets in section 4.5, we were unable to find significant differences. Furthermore, in the data preprocessing pipeline, many of these differences are being reduced due to standardization and filtering. This is not to say that signal differences cannot exist in the data fed to the model, as we lack domain knowledge to determine actual differences. This is a reason why a pre-training pipeline is important, as we have no domain knowledge to identify and extract important features, which the autoencoder does automatically. Our balance of $\approx 33\%$ stress data was somewhat arbitrary since it was decided to ensure an identical distribution to the first dataset we looked at; the DTU dataset. At no point did we try other event/nonevent distributions. This made comparisons with Lønfeldt et al. 2023 difficult, as their distribution was 50% event. Additionally, we only down-sampled non-events and did not try to up-sample events. If we had up-sampled the event data, we would have had more data, and thus data scarcity could have been a smaller problem. Furthermore, this is supported by [Wongvorachan, He, and Bulut 2023] for a moderately imbalanced dataset ($\approx 30\%$) oversampling improved the performance, however when the data is very imbalanced a hybrid resampling technique proves superior. [Khushi et al. 2021] found similar results when using a moderately imbalanced dataset; random oversampling is proven superior compared to no resampling and 22 other resampling methods. Thus, instead of downsampling to the current distribution using the hybrid resampling or random oversampling technique on all observations could provide better results.

The WESAD, ROAD, and DTU datasets were all captured in a controlled setting, whereas the Wrist Angel and ADARP datasets were collected in the wild. In the WA study, the participants may sometimes have accidentally tagged an event. The E4 wristband may have had down periods, with broken sensors or poor calibration, etc., as observed with the temperature data in fig. 4.11d. Furthermore, the action of tagging could become an obsession for OCD patients or could serve as a relaxing action to reduce symptoms, leading to a larger number of false tags. All these facts make for poorer data quality, where the differences between an event and a non-event are less, making accurate predictions harder. Furthermore, the open-source datasets used are not data from OCD patients who experience stress due to OCD symptoms. Therefore, there may be some differences in how stress events and OCD episodes affect physiological signals.

Simulated Data
As described in section 5.1, we experimented with different simulated data for adding to our model to increase performance.
When including the simulated data with the final model at the end of our work, it improved performance as seen in table 6.23. A significant increase in performance is especially seen in the more noisy data that was created with the fragmented method. This does support our initial hypothesis that including more noisy simulated data will improve the model, because it has seen more varied signals and, therefore, is better prepared for predicting new data. This is further supported by our literature review where [Minor et al. 2020; Alkhalifah, H. Wang, and Ovcharenko 2022] both found that adding noise to the simulated data improved their model performance. The reason our initial tests of the fragmented simulated data did not improve performance might be because it was tested only on one dataset. Whereas with our last tests, which showed good results all four datasets were used. Unfortunately, due to time restrictions, and this finding coming rather late in the project, we did not get to test the inclusion of the fragmented simulated data on the Wrist Angel dataset.

7.2 Model

From sections 6.1 and 6.2 it is clear that a lot of time and effort was spent to find a good model. Interestingly, the more complex model architectures, like a transformer-based architecture, performed worse, most likely because of the limited amount of data, which was especially true at the time of testing these models. This is somewhat similar to the results in [Deznabi and Fiterau 2023], where for the WESAD dataset [Schmidt et al. 2018] transformer and LSTM-based networks performed worse compared to convolutional-based networks. For other datasets performance was more similar, suggesting that the more complex models can work, but are less reliable across datasets. However, complex models also took significantly longer time to run, making them unsuitable for wearables with far less computational power.

The uncommon addition of a separate model for each included frequency (inspired by [Deznabi and Fiterau 2023]) seemed to provide superior performance, suggesting that, it is a disadvantage using only up- or downsampling signals. Unfortunately, we were unable to find literature regarding this. We chose not to include a CNN (or any other complex) network as the decoder, as this would have added too much complexity to the model. A small peculiarity of the network is the adaptive max pooling, as this converts all inputs regardless of lengths (smallest $240 = 60 \cdot 4$ and largest $19200 = 300 * 64$) to 50, this is a large reduction in data dimensionality. However, as seen in table 6.19 this addition gives superior performance. We frequently saw the model overfitting to the training data, especially in the personal training scenarios fig. 6.11. Several strategies were tested to combat this issue, like higher drop rates and using a simpler model. However, the simpler models yielded worse performance (table 6.13), indicating some complexity in the data, and thus also the need for a complex model. This highlights the trade-off between the required complexity and the required amount of data, to achieve good performance without overfitting. In general, we saw good model performance before applying the model on the WA dataset. The ADARP[1], DTU and ROAD datasets did not have a comparable metric for stress prediction performance, as stress prediction was not the main focus of these datasets. For WESAD [Schmidt et al. 2018] they had an F1-score of 86% with a random forest model (RF), meaning our model outperformed theirs (see table 6.25). For

[1]The authors of this article achieved an F1-score of 0.98 [Sah et al. 2022], which seems odd, especially compared to our results. Thus we don't think the two scenarios are comparable.

the in-the-wild dataset, ADARP, the F1-score was 0.25 when pre-training (table 6.25), which was significantly worse than the other datasets, indicating that our model can't properly handle in-the-wild data.

7.3 Transfer Learning

The essence of transfer learning is to train a model using similar data for a similar task such that it can learn the underlying representation. Our pre Wrist Angel results indicated mixed results for the fine-tuning of the pre-trained model. Importantly, the usage of the pre-trained model, either by directly predicting or fine-tuning, was always better compared to training solely on the test set (table 6.25). The mixed success of the application of transfer learning also mirrored the results of WA section 6.9.4, which also did not indicate a clear performance gain, when fine-tuning the pre-trained model. However, the performance did not decrease either. The inconsistent success of transfer learning indicates the existence of differences between datasets, as the results should otherwise be similar and probably better. This is confirmed by another study on using transfer learning to predict the strength of concrete columns [Pak and Paal 2022], which states that the success of transfer learning is mostly affected by the physical relationship between the source and target domain. Despite this, it can still be useful knowing that using a pre-trained model (even though there are differences between the source and target domains) does not significantly hurt performance, as many open-source pre-trained models exist. These models can then be used in this specific domain, with the performance being indicative of the level of performance that can be expected to be obtained.

7.4 Factors

Activity

From tables 6.30 to 6.33 it seems removing observations classified as physical activity does not affect the overall performance of the model. Since we do not know whether the observations classified as physical activity are true or false, we cannot assess how accurate the classification is. We know from the activity lab recordings made as part of the Wrist Angel study, that our activity classification made a perfect split between activity and non-activity fig. 6.6. However, from fig. 6.13 we did not see a greater number of false positives for observations classified as activity, which should have been the case if activity classification worked well. But, as we also know from the WEEE dataset, for instance, cycling is not as visible in the acceleration data as running. Hence the participants in the Wrist Angel study could have performed different physical activities, where the wrist is not in motion, therefore these would not be captured by the activity classification. Thus, a solution could be to use a more complex model that also considers the physiological signals. On the other hand, this again increases complexity, which we have generally tried to avoid.

Window Length

As we saw from section 6.9.3 the 1-minute window length performed the best. Most of the initial models were performed with a 5-minute window, and thus the model should be biased toward this window length. Therefore, this could indicate that only a small amount of proceeding data on the event has relevant information. But it could also be that a 5-minute window contains too much noise, and thus the model is unable to extract the relevant information. Schmidt et al. showed good results in their study [Schmidt et al. 2018], which was conducted on the WESAD dataset using a 1-minute window length. This window length was selected based on [Kreibig 2010]. Kreibig saw that while watching embarrassing videos, the participants' HR rose during the first minute, but returned to baseline in the second minute. This could indicate that some physiological signals would

only show a change during a small window of time, thus the inclusion of the extra duration could dilute the event.

Lead prediction time
Based on tables 6.33 and 6.35 we can see that a shorter lead prediction time (LPT) is not necessarily preferable to a longer LPT. This is similar to what we saw from the initial ADARP tests in table 6.20, where a 2-minute LPT seemed to be best. This could indicate that OCD events (and stress episodes in the ADARP case) are revealed in physiological signals a couple of minutes ahead of the OCD episode rather than just before the episode. But this is opposite to our findings from the window length, as this would mean the event is within the 5-min window, and not the 1-min window, meaning the 5-min window should perform better. Thus it is more likely, that there is an issue with the collection of the data. It appears participants may not tag when they first experience the OCD event but instead tag at some time during or after their episode. This could mean that a 0-minute LPT sometimes results in the observation being sampled after the event. In contrast, for a longer LPT, the sampled observation would be just before or during the OCD event. From fig. 6.15 we saw that LPT had different effects on model performance for different participants, seemingly strengthening this claim. However, the model performance is somewhat flawed, which is probably why we see a uniform influence of LPT for some participants. The flawed performance indicates that the model's precision is too poor to assess the bias and a superior LPT. Thus it would mean that there is an 83% $(5/6)$ probability that a non-zero LPT is best by chance.

7.5 Model Performance on Wrist Angel

The highest F1-score of our model applied to the Wrist Angel dataset found is only 0.50 in the random pre-trained on the 1-minute Wrist Angel dataset, with a 3-minute lead prediction time, without activity classification (or 5-minute LPT with activity classification). Therefore, this is not a good reliable model to predict OCD events, and it brings many of the results into question. As discussed there could be multiple reasons for the inadequate performance. For one, although there are no significant visual differences between datasets and physiological signals, they can still exist. In addition, the lack of good transfer learning further supports this. The data quality for WA is most likely bad, especially regarding the tagging. It should not be the case that different non-zero LPTs are best. We did see good performance from our model on the other datasets, although the only other in-the-wild dataset, ADARP, did seem to indicate the applicability problems we faced with our model.

Given the lacking results of transfer learning and the general applicability of our model, this begs the question if a greater emphasis should have been on the Wrist Angel data through strategies such as self-supervised learning, data cleaning, and feature extraction. The difference in the amount of available data for OCD and stress is vast. When you furthermore, add the fact that data is obtained from adolescents, it becomes impossible to find data similar to the WA data. This problem is not limited to this dataset, it exists in many domains, especially regarding labeled data in healthcare. Therefore, applying data/models from a similar area can be beneficial. Given a low resource domain, model performance will likely be mediocre, meaning any performance boost is significant. As shown in section 6.9.4 it seems the addition of a pre-trained model did not worsen performance.

It is unclear to what degree hyperparameter tuning could have helped performance, and this is something that would be interesting to look at in the future. Instead of seeing our pre-trained model as a starting point or as a baseline, it could then be adjusted to the WA

data. This is, of course, counter to our SAP, as the point was to be unbiased by data knowledge.

As was shown in section 6.2 our self-supervised strategy did not seem to improve performance. However, this could be due to the chosen tasks not having good transferability to the stress classification task. For the contrastive task, it may be the case that the model doesn't need to recognize stress in the signals, but just the difference between them, to classify if they belong to the same person. Additionally, it might ”confuse” the model, as a stress signal and a non-stress signal from the same person, would be classified as the same. In contrast, for the classification task, they are different, ie. learning the model to ignore stress.

7.6 Evaluation Method

Throughout this report, we have reported both accuracy and F1-score as a metric of model performance. While accuracy is more intuitive the metric does not account for imbalanced data, which we are working with. This is why we mainly used the F1-score, which is also recommended for binary classification in other studies such as [Hossin and Sulaiman 2015]. From table 6.38 this problem is clear as maximizing event threshold for accuracy, comes at the cost of F1-score as the model predicts for the over-represented class (non-stress). A problem with the interpretability of the F1-score is the balance between precision and recall, as different users might weigh the two differently, with valid reasons for each. In this case, an F_β-score could be used, to individually adjust β as to how much more important recall is than precision. Alternatively, recall and precision are as mentioned other variables that can be ”optimized” by the threshold for the individual user.

7.7 Comparison to Previous Wrist Angel Study

We generally do not achieve on-par performance when comparing our results to the previous study on the Wrist Angel data [Lønfeldt et al. 2023]. Lønfeldt et al. reported an average F1-score from their random cross-validation strategy to be just under 0.6, and an average accuracy of around 0.7. Whereas in our best-performing scenario according to the F1-score, the average was around 10 percentage points below, namely 0.50. Similarly, according to accuracy, the best average was at 0.64. It should, however, be noted that the two studies are not directly comparable as the data used is not identical even though they come from the same dataset. This is because the pre-processing differs, and so does the selection of observations, which have been included in training and predicting. Probably of most significance is the different balances of event vs. nonevent, as Lønfeldt et al. uses a 50-50 split, compared to our 1/3 event split. Furthermore, the methods vary significantly in different aspects, such as Lønfeldt et al. using extracted features from the observations, which are in other studies proved to be effective [Campanella et al. 2023]. These features are however manually selected, which we avoid by using an autoencoder with time series data. Moreover, our model was created based on open-source stress data, without knowledge of the Wrist Angel data. Conversely, the model used in [Lønfeldt et al. 2023] is created and trained specifically on the Wrist Angel data. Nevertheless, similar trends are seen regarding data split strategies. As reported in table 6.37 in section 6.9.4, the personalized splits usually perform worse than the random splits. The trend found here corresponds to that of [Lønfeldt et al. 2023] since they also report worse performance for their participant-based splits than their random split.

8 Conclusion

This study explored the potential of using deep learning models and transfer learning from stress data to predict OCD episodes. The main novelty of this study is the utilization of real-world and simulated data, thereby addressing the issue of data scarcity in the medical domain. Specifically, building upon the Wrist Angel (WA) dataset containing 8 weeks of daily physiological data from children diagnosed with OCD. We investigated and tested if using a pre-trained model in stress events can help predict OCD events. During this, different factors were further explored and elaborated on.

First, we conducted an extensive evaluation of different model architectures, including CNN, LSTM, attention, and Transformer-based models. From these CNN was found to have the best performance and was thus chosen to be applied to the WA data. Meanwhile, we simulated data using our fragmented method, by simulating short 3-second intervals and later concatenating them together. It was done in an attempt to improve model performance, which unfortunately was not initially the case, but has shown promising results towards the end of the project.

Thereafter, we pre-trained the model using four different open-source datasets; ADARP, DTU, ROAD, and WESAD. Here, we applied extensive hyperparameter tuning, such as the number of layers, kernel size, loss function, etc. This led to generally good performance when tested on open-source stress datasets. Especially for the WESAD dataset, we achieved a high F1-score of 0.95, which is superior to a previous study.

Lastly, we tested the pre-trained models on the WA dataset. The good performance seen on the open-source datasets did not carry over to the Wrist Angel dataset. 120 scenarios were tested where we focused on assessing the impact of four different factors: window length, lead prediction time, activity classification, and application method. In particular, five application methods were used. Directly predicting using the pre-trained model, fine-tuning the pre-trained model with a random split, fine-tuning the pre-trained model with a personalized split, the model solely trained on the WA dataset with a random split, and finally, the model solely trained on the WA dataset with a personalized split. Comparing the application methods, we found no performance gain in using pre-trained models, and that the personalized split performed significantly worse than the random split when taking F1-score into account. As for lead prediction time (LPT) a non-zero LPT seemed superior, and for the inclusion of an activity classification filter, no significant improvement in performance was gained. Furthermore, the results indicated that a shorter window length of one minute is beneficial to a length of five minutes. However, all model-results should be taken with caution given the lack of performance.

In summary, we have not increased performance in predicting OCD events from the Wrist Angel study, but we have shown potential for utilizing simulated data and transfer learning, mostly due to the results from open-source datasets.

9 Future Work

Future studies could aim at increasing the predictability of OCD events in the Wrist Angel (WA) dataset by creating, tuning, and training the model on the WA dataset. Because we in our study have not seen great improvements from using a model pre-trained on different datasets, it might improve performance if the model is hyperparameter-tuned to the WA dataset. Likewise, the model might benefit from self-supervised learning methods adjusted and trained on the WA dataset.

Due to the late finding of an increase in model performance, by including the fragmented simulated data, it would be interesting to test the inclusion of simulated data on the WA dataset. It would likely increase performance, given that the open-source datasets benefited from including the simulated data. Even ADARP, which is also an in-the-wild dataset, sees an increase in performance. Additionally, as described in the discussion, the simulated data had multiple limitations. Future work could explore other simulation techniques. First, it might be possible to develop a physiological model capable of simulating physiological signals (BVP, EDA, HR, and TEMP) concurrently, ensuring their correlation mimics that of the human body. We found no such models in our literature review. Secondly, as described in our literature review Generative Adversarial Networks (GANs) can be utilized to generate synthetic data given an adequate amount of initial data and computational resources.

Data limitations regarding quality and the amount of data are an issue. With the current WA data, it could be interesting to include data from the lab recordings. Because, here the participants have been in a controlled environment and experienced being stressed by inducing OCD events. This could act as a baseline for a model, with more certainty about the time of the event. Additionally, data is available from the control group. It could give another insight into how physiological signals behave in adolescents diagnosed with OCD, by comparing them to adolescents with no diagnosis. Furthermore, data from the parents have also been collected. They have tagged every time they noticed their child being distressed by OCD symptoms. Hence by exploring this data, tags of the child and their parent could be compared to get better knowledge of the OCD event. Additionally, the physiological data from the parents can be used to find differences and similarities between physiological signals in children and adults. From these added data sources relevant self-supervised learning tasks can be created, like a classification task distinguishing between the adolescents with OCD and the control group. Or contrastive learning with the pretext task of recognizing OCD events in a lab environment or in-the-wild.

Finally, additional in-the-wild data collection could further develop our study, and help to determine the real-world applicability of predicting OCD events from live wristband-measured physiological data. This would likely also help to improve the performance of applying a deep learning model. In general, regarding the collection of new data, data quality is also an important consideration. For instance, knowing the exact start time and duration of the OCD symptoms could improve the data quality and thereby the performance.

Bibliography

Abramowitz, Jonathan S., Michael G. Wheaton, and Eric A. Storch (2008). "The status of hoarding as a symptom of obsessive–compulsive disorder". In: *Behaviour Research and Therapy* 46.9, pp. 1026–1033. ISSN: 0005-7967. DOI: https://doi.org/10.1016/j.brat.2008.05.006. URL: https://www.sciencedirect.com/science/article/pii/S0005796708001253.

Adaloglou, Nikolas (May 12, 2022). *BYOL tutorial: self-supervised learning on CIFAR images with code in Pytorch*. AI Summer. URL: https://theaisummer.com/byol/ (visited on 01/24/2024).

Alinia, Parastoo et al. (July 2021). "Associations Between Physiological Signals Captured Using Wearable Sensors and Self-reported Outcomes Among Adults in Alcohol Use Disorder Recovery: Development and Usability Study". In: *JMIR Formative Research* 5.7, e27891. ISSN: 2561-326X. DOI: 10.2196/27891.

Alkhalifah, Tariq, Hanchen Wang, and Oleg Ovcharenko (Dec. 2022). "MLReal: Bridging the gap between training on synthetic data and real data applications in machine learning". en. In: *Artificial Intelligence in Geosciences* 3, pp. 101–114. ISSN: 26665441. DOI: 10.1016/j.aiig.2022.09.002.

Azizi, Shekoofeh et al. (Oct. 2021). "Big Self-Supervised Models Advance Medical Image Classification". en. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Montreal, QC, Canada: IEEE, pp. 3458–3468. ISBN: 978-1-66542-812-5. DOI: 10.1109/ICCV48922.2021.00346. URL: https://ieeexplore.ieee.org/document/9710396/.

B, Harikrishnan N (2019). "Confusion Matrix, Accuracy, Precision, Recall, F1 Score". In: *Medium*. Accessed: 2024-01-02. URL: https://medium.com/analytics-vidhya/confusion-matrix-accuracy-precision-recall-f1-score-ade299cf63cd.

Benjamini, Yoav and Yosef Hochberg (1995). "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 57.1, pp. 289–300. ISSN: 00359246. URL: http://www.jstor.org/stable/2346101 (visited on 01/25/2024).

Boderskov, Max (2023). *Definitionen af HRV*. Accessed: 2023-10-04. URL: https://www.loebeshop.dk/inspiration/guide-hvad-er-hrv.

Boucsein, Wolfram (2012). *Electrodermal Activity*. Springer New York, NY. ISBN: 978-1-4614-1125-3. DOI: 10.1007/978-1-4614-1126-0. URL: https://doi-org.proxy.findit.cvt.dk/10.1007/978-1-4614-1126-0.

Bustos, Cristina et al. (Sept. 2021). "Predicting Driver Self-Reported Stress by Analyzing the Road Scene". In: arXiv:2109.13225. arXiv:2109.13225 [cs]. URL: http://arxiv.org/abs/2109.13225.

Campanella, Sara et al. (Jan. 2023). "A Method for Stress Detection Using Empatica E4 Bracelet and Machine-Learning Techniques". In: *Sensors* 23.7. Number: 7 Publisher: Multidisciplinary Digital Publishing Institute, p. 3565. ISSN: 1424-8220. DOI: 10.3390/s23073565. URL: https://www.mdpi.com/1424-8220/23/7/3565 (visited on 09/18/2023).

CDC (2022). *How is BMI calculated?* Accessed: 2023-10-04. URL: https://www.cdc.gov/healthyweight/assessing/bmi/adult_bmi/index.html#Interpreted.

Chen, Shiyi, Zhe Feng, and Xiaolian Yi (2017). "A general introduction to adjustment for multiple comparisons." In: *Journal of thoracic disease* 9 6, pp. 1725–1729. URL: https://api.semanticscholar.org/CorpusID:26331244.

Chen, Wei, Shixin Zheng, and Xiao Sun (2021). "Introducing MDPSD, a Multimodal Dataset for Psychological Stress Detection". en. In: *Big Data*. Ed. by Hong Mei et al. Vol. 1320. Communications in Computer and Information Science. Singapore: Springer Singapore, pp. 59–82. DOI: 10.1007/978-981-16-0705-9_5. URL: http://link.springer.com/10.1007/978-981-16-0705-9_5.

Chollet, Francois et al. (2015). *Keras*. URL: https://github.com/fchollet/keras.

Deznabi, Iman and Madalina Fiterau (June 2023). "MultiWave: Multiresolution Deep Architectures through Wavelet Decomposition for Multivariate Time Series Prediction". In: arXiv:2306.10164. arXiv:2306.10164 [cs, eess]. URL: http://arxiv.org/abs/2306.10164.

DMI (2023). *Vejrarkiv*. [Online; accessed 4-October-2023]. URL: https://www.dmi.dk/vejrarkiv/.

E4 sensors (2023). en-US. URL: https://www.empatica.com/en-int/research/e4/.

E4 wristband technical specifications (2024). en-US. Accessed: 02-01-2024. URL: https://support.empatica.com/hc/en-us/articles/202581999-E4-wristband-technical-specifications.

Edwards, Allen L (1948). "Note on the "correction for continuity" in testing the significance of the difference between correlated proportions". In: *Psychometrika* 13.3, pp. 185–187.

Empatica (2023a). *Data export and formatting from E4 connect*. en-US. Accessed: 2023-11-21. URL: https://support.empatica.com/hc/en-us/articles/201608896-Data-export-and-formatting-from-E4-connect-.

– (2023b). *E4 data - BVP expected signal*. Accessed: 2023-10-02. URL: https://support.empatica.com/hc/en-us/articles/360029719792-E4-data-BVP-expected-signal.

– (2023c). *E4 Wristband*. Accessed: 2023-11-21. URL: https://e4.empatica.com/e4-wristband.

Empatica Legal & Compliance (2024). en. Accessed: 02-01-2024. URL: https://www.empatica.com/legal.

Gambera, Matteo (Feb. 2021). "How to extract Features from Signals". en. In: *Medium*. URL: https://matteogambera.medium.com/how-to-extract-features-from-signals-15e7db225c15.

Garbarino, M. et al. (Nov. 2014). "Empatica E3 - A wearable wireless multi-sensor device for real-time computerized biofeedback and data acquisition". In: *2014 EAI 4th International Conference on Wireless Mobile Communication and Healthcare (Mobihealth)*, pp. 39–42. DOI: 10.1109/MOBIHEALTH.2014.7015904.

Gashi, Shkurta et al. (Sept. 2022). "A multidevice and multimodal dataset for human energy expenditure estimation using wearable devices". In: *Scientific Data* 9. DOI: 10.1038/s41597-022-01643-5.

Giannakakis, Giorgos et al. (Jan. 2022). "Review on Psychological Stress Detection Using Biosignals". In: *IEEE Transactions on Affective Computing* 13.1, pp. 440–460. ISSN: 1949-3045. DOI: 10.1109/TAFFC.2019.2927337.

GMTLondon (2023). *Buying Watches for Children*. Accessed: 2023-10-04. URL: https://gmtlondon.com/guides/buying-watches-children.

Grill, Jean-Bastien et al. (Sept. 10, 2020). *Bootstrap your own latent: A new approach to self-supervised Learning*. arXiv: 2006.07733[cs,stat]. URL: http://arxiv.org/abs/2006.07733 (visited on 01/24/2024).

Group, Medical Device Coordination (Oct. 2021). *MDCG 2021-24 Guidance on classification of medical devices*. URL: https://health.ec.europa.eu/system/files/2021-10/mdcg_2021-24_en_0.pdf.

Haouij, Neska El et al. (2018). "AffectiveROAD System and Database to Assess Driver's Attention". In: *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*. SAC '18. Pau, France: Association for Computing Machinery, pp. 800–803. ISBN:

9781450351911. DOI: 10.1145/3167132.3167395. URL: https://doi.org/10.1145/3167132.3167395.
Harris, Charles R. et al. (Sept. 2020). "Array programming with NumPy". In: *Nature* 585.7825, pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.
Hipp, Deb (2023). *Normal Resting Heart Rate By Age (Chart)*. Accessed: 2023-10-04. URL: https://www.forbes.com/health/healthy-aging/normal-heart-rate-by-age/.
Hossin, Mohammad and Md Nasir Sulaiman (2015). "A review on evaluation metrics for data classification evaluations". In: *International journal of data mining & knowledge management process* 5.2, p. 1.
Hu, Rui, Jie Chen, and Li Zhou (2022). "A transformer-based deep neural network for arrhythmia detection using continuous ECG signals". en. In: *Computers in Biology and Medicine* 144, p. 105325. DOI: 10.1016/j.compbiomed.2022.105325. URL: https://linkinghub.elsevier.com/retrieve/pii/S0010482522001172.
Hudson, Jennifer L. and Ronald M. Rapee (2001). "Parent-child interactions and anxiety disorders: An observational study". English. In: *Behaviour Research and Therapy* 39.12, pp. 1411–1427. ISSN: 0005-7967. DOI: 10.1016/S0005-7967(00)00107-8.
IBM (2023). *What are convolutional neural networks?* Accessed: 2023-11-29.
Initiative, NHLBI Obesity Education (2000). *The Practical Guide Identification, Evaluation, and Treatment of Overweight and Obesity in Adults*. National Institutes of Health. ISBN: 978-1-4614-1125-3. DOI: 10.1007/978-1-4614-1126-0. URL: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.nhlbi.nih.gov/files/docs/guidelines/prctgd_c.pdf.
Ioffe, Sergey and Christian Szegedy (Mar. 2, 2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. arXiv: 1502.03167[cs]. URL: http://arxiv.org/abs/1502.03167 (visited on 01/09/2024).
Jaadi, Zakaria (2024). *When and Why to Standardize Your Data*. Accessed 16-01-2024. URL: https://builtin.com/data-science/when-and-why-standardize-your-data.
K, Prabhavathi et al. (Aug. 2014). "Role of Biological Sex in Normal Cardiac Function and in its Disease Outcome – A Review". In: *J Clin Diagn Res* 8. DOI: 10.7860/JCDR/2014/9635.4771.
Khushi, Matloob et al. (2021). "A Comparative Performance Analysis of Data Resampling Methods on Imbalance Medical Data". en. In: *IEEE Access* 9, pp. 109960–109975. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3102399.
Kirschbaum, Clemens, Karl-Martin Pirke, and Dirk Hellhammer (Feb. 1993). "The 'Trier Social Stress Test' – A Tool for Investigating Psychobiological Stress Responses in a Laboratory Setting". In: *Neuropsychobiology* 28, pp. 76–81. DOI: 10.1159/000119004.
Kobayashi, Yutaka and Shin-ichi Tanabe (2013). "Development of JOS-2 human thermoregulation model with detailed vascular system". In: *Building and Environment* 66, pp. 1–10. ISSN: 0360-1323. DOI: https://doi.org/10.1016/j.buildenv.2013.04.013. URL: https://www.sciencedirect.com/science/article/pii/S0360132313001182.
Kreibig, Sylvia D. (July 2010). "Autonomic nervous system activity in emotion: A review". en. In: *Biological Psychology* 84.3, pp. 394–421. ISSN: 03010511. DOI: 10.1016/j.biopsycho.2010.03.010.
Kumar, Amit Krishan et al. (2022). "Deep learning for predicting respiratory rate from biosignals". In: *Computers in Biology and Medicine* 144, p. 105338. DOI: https://doi.org/10.1016/j.compbiomed.2022.105338. URL: https://www.sciencedirect.com/science/article/pii/S0010482522001305.

Längkvist, Martin, Lars Karlsson, and Amy Loutfi (June 2014). “A review of unsupervised feature learning and deep learning for time-series modeling”. In: *Pattern Recognition Letters* 42, pp. 11–24. ISSN: 0167-8655. DOI: 10.1016/j.patrec.2014.01.008.

Li, Xiaomin, Anne Hee Hiong Ngu, and Vangelis Metsis (June 2022). “TTS-CGAN: A Transformer Time-Series Conditional GAN for Biosignal Data Augmentation”. In: arXiv:2206.13676. arXiv:2206.13676 [cs]. URL: http://arxiv.org/abs/2206.13676.

Lønfeldt, Nicole Nadine et al. (2023). “Predicting Obsessive-Compulsive Disorder Episodes in Adolescents using a Wearable Biosensor - A Wrist Angel Feasibility Study”. In.

Lopez-Martinez, Daniel, Neska El-Haouij, and Rosalind Picard (July 19, 2019). *Detection of Real-world Driving-induced Affective State Using Physiological Signals and Multi-view Multi-task Machine Learning*. arXiv: 1907.09929[cs,stat]. URL: http://arxiv.org/abs/1907.09929 (visited on 11/07/2023).

Makowski, Dominique et al. (Feb. 2021). “NeuroKit2: A Python toolbox for neurophysiological signal processing”. In: *Behavior Research Methods* 53.4, pp. 1689–1696. DOI: 10.3758/s13428-020-01516-y. URL: https://doi.org/10.3758%2Fs13428-020-01516-y.

Martín Abadi et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: https://www.tensorflow.org/.

Matte, Michelle (2019). *What Is a Realistic BMI for Someone Athletic?* Accessed: 2023-10-04. URL: https://www.livestrong.com/article/395464-what-is-a-realistic-bmi-for-someone-athletic/.

McClenaghan, Elliot (2024). *Mann-Whitney U Test: Assumptions and Example*. Accessed 15-01-2024. URL: https://www.technologynetworks.com/informatics/articles/mann-whitney-u-test-assumptions-and-example-363425.

McNemar, Quinn (1947). “Note on the sampling error of the difference between correlated proportions or percentages”. In: *Psychometrika* 12.2, pp. 153–157.

McSharry, P.E. et al. (2003). “A dynamical model for generating synthetic electrocardiogram signals”. In: *IEEE Transactions on Biomedical Engineering* 50.3, pp. 289–294. DOI: 10.1109/TBME.2003.808805.

Merritt, Rick (Mar. 25, 2022). *What Is a Transformer Model?* NVIDIA Blog. URL: https://blogs.nvidia.com/blog/what-is-a-transformer-model/ (visited on 01/09/2024).

Mind (2023). *How can I help myself with OCD?* Accessed 19-01-2024. URL: https://www.mind.org.uk/information-support/types-of-mental-health-problems/obsessive-compulsive-disorder-ocd/self-care-for-ocd/.

Minor, Eric N. et al. (2020). “End-to-end machine learning for experimental physics: using simulated data to train a neural network for object detection in video microscopy”. en. In: *Soft Matter* 16.7, pp. 1751–1759. ISSN: 1744-683X, 1744-6848. DOI: 10.1039/C9SM01979K.

Montero Quispe, Kevin G. et al. (Nov. 2022). “Applying Self-Supervised Representation Learning for Emotion Recognition Using Physiological Signals”. en. In: *Sensors* 22.23, p. 9102. ISSN: 1424-8220. DOI: 10.3390/s22239102.

NVIDIA (2024a). *Long Short-Term Memory (LSTM)*. NVIDIA Developer. URL: https://developer.nvidia.com/discover/lstm (visited on 01/09/2024).

– (2024b). *What is a Convolutional Neural Network?* NVIDIA Data Science Glossary. URL: https://www.nvidia.com/en-us/glossary/data-science/convolutional-neural-network/ (visited on 01/09/2024).

Olesen, Kristoffer Vinther, Sneha Das, et al. (2023). “Predicting Obsessive-Compulsive Disorder Events in Children and Adolescents from BioSignals In-the-Wild - An Analysis Plan”. In.

Olesen, Kristoffer Vinther, Nicole Nadine Lønfeldt, et al. (Nov. 2023). “Predicting Obsessive-Compulsive Disorder Events in Children and Adolescents in the Wild Using a Wearable

Biosensor (Wrist Angel): Protocol for the Analysis Plan of a Nonrandomized Pilot Study". In: *JMIR Res Protoc* 12, e48571. ISSN: 1929-0748. DOI: 10.2196/48571. URL: http://www.ncbi.nlm.nih.gov/pubmed/37962931.

Pak, Hongrak and Stephanie German Paal (2022). "Evaluation of transfer learning models for predicting the lateral strength of reinforced concrete columns". In: *Engineering Structures* 266, p. 114579. ISSN: 0141-0296. DOI: https://doi.org/10.1016/j.engstruct.2022.114579. URL: https://www.sciencedirect.com/science/article/pii/S0141029622006824.

Panasonic (2023). *The key to comfortable living: perfect room temperature and humidity*. [Online; accessed 4-October-2023]. URL: https://www.panasonic.com/global/hvac/nanoe/stories/comfortable-temperature-and-humidity.html.

Paszke, Adam et al. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Patel N Durland J, Makaryus AN (Sept. 2022). "Physiology, Cardiac Index". In: *StatPearls Publishing*. URL: https://www.ncbi.nlm.nih.gov/books/NBK539905/.

Potrimba, Petru (2023). *What is a Convolutional Neural Network?* Accessed 16-01-2024. URL: https://blog.roboflow.com/what-is-a-convolutional-neural-network/.

PyTorch (2023). *Conv1d — PyTorch 2.1 documentation*. CONV1D. URL: https://pytorch.org/docs/stable/generated/torch.nn.Conv1d.html (visited on 01/05/2024).

– (2024a). *AdaptiveMaxPool1d — PyTorch 2.1 documentation*. URL: https://pytorch.org/docs/stable/generated/torch.nn.AdaptiveMaxPool1d.html (visited on 01/09/2024).

– (2024b). *BatchNorm1d — PyTorch 2.1 documentation*. URL: https://pytorch.org/docs/stable/generated/torch.nn.BatchNorm1d.html (visited on 01/09/2024).

– (2024c). *Dropout — PyTorch 2.1 documentation*. URL: https://pytorch.org/docs/stable/generated/torch.nn.Dropout.html (visited on 01/09/2024).

– (2024d). *LeakyReLU — PyTorch 2.1 documentation*. URL: https://pytorch.org/docs/stable/generated/torch.nn.LeakyReLU.html (visited on 01/09/2024).

Rácz, Anita, Dávid Bajusz, and Károly Héberger (2021). "Effect of Dataset Size and Train/Test Split Ratios in QSAR/QSPR Multiclass Classification". In: *Molecules* 26.4. ISSN: 1420-3049. DOI: 10.3390/molecules26041111. URL: https://www.mdpi.com/1420-3049/26/4/1111.

Rallabandi, Srikari (2023). *Activation functions: ReLU vs. Leaky ReLU*. Accessed 16-01-2024. URL: https://medium.com/mlearning-ai/activation-functions-relu-vs-leaky-relu-b8272dc0b1be.

Raschka, Sebastian (Apr. 2018). "MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack". In: *The Journal of Open Source Software* 3.24. DOI: 10.21105/joss.00638. URL: https://joss.theoj.org/papers/10.21105/joss.00638.

– (2024). *mcnemar: McNemar's test for classifier comparisons*. Accessed: 2024-01-30. URL: https://rasbt.github.io/mlxtend/user_guide/evaluate/mcnemar/.

Rautaharju, P. and F. Rautaharju (2007). "Heart Rate and Heart Rate Variability". In: *Investigative Electrocardiography in Epidemiological Studies and Clinical Trials*. London: Springer London, pp. 45–67. ISBN: 978-1-84628-481-6. DOI: 10.1007/978-1-84628-481-6_4. URL: https://doi.org/10.1007/978-1-84628-481-6_4.

Robinson, Lisa (1990). "Stress and Anxiety". In: *Nursing Clinics of North America* 25.4, pp. 935–943. ISSN: 0029-6465. DOI: https://doi.org/10.1016/S0029-6465(22)02991-7. URL: https://www.sciencedirect.com/science/article/pii/S0029646522029917.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *CoRR* abs/1505.04597. arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

Sah, Ramesh Kumar et al. (June 2022). “ADARP: A Multi Modal Dataset for Stress and Alcohol Relapse Quantification in Real Life Setting”. In: arXiv:2206.14568. arXiv:2206.14568 [cs, eess]. URL: http://arxiv.org/abs/2206.14568.

Schmidt, Philip et al. (2018). “Introducing WESAD, a Multimodal Dataset for Wearable Stress and Affect Detection”. In: *Proceedings of the 20th ACM International Conference on Multimodal Interaction*. ICMI ’18. Boulder, CO, USA: Association for Computing Machinery, pp. 400–408. ISBN: 9781450356923. DOI: 10.1145/3242969.3242985. URL: https://doi.org/10.1145/3242969.3242985.

Scott, Elizabeth (2021). *All About Acute Stress*. Accessed: 2023-09-27. URL: https://www.verywellmind.com/all-about-acute-stress-3145064.

Skat-Rørdam, Harald Vilhelm et al. (2023). *Applying Pre-Trained Deep-Learning Model on Wrist Angel Data – An Analysis Plan*. arXiv: 2312.09052 [stat.AP].

Stolwijk, J.A.J. et al. (1971). *A Mathematical Model of Physiological Temperature Regulation in Man*. NASA contractor report. National Aeronautics and Space Administration. URL: https://books.google.dk/books?id=UdgTLYu0JBEC.

Symptoms - Obsessive compulsive disorder (OCD) (Feb. 2021). en. URL: https://www.nhs.uk/mental-health/conditions/obsessive-compulsive-disorder-ocd/symptoms/.

Takahashi, Yoshito et al. (2021). “Thermoregulation model JOS-3 with new open source code”. In: *Energy and Buildings* 231, p. 110575. ISSN: 0378-7788. DOI: https://doi.org/10.1016/j.enbuild.2020.110575. URL: https://www.sciencedirect.com/science/article/pii/S0378778820333612.

Taylor, Sebastian (2024). *Nonparametric Tests*. Accessed 15-01-2024. URL: https://corporatefinanceinstitute.com/resources/data-science/nonparametric-tests/.

Theunissen, Carl Daniel et al. (2021). “One-Dimensional Convolutional Auto-Encoder for Predicting Furnace Blowback Events from Multivariate Time Series Process Data—A Case Study”. In: *Minerals* 11.10. ISSN: 2075-163X. DOI: 10.3390/min11101106. URL: https://www.mdpi.com/2075-163X/11/10/1106.

Tremblay, Jonathan et al. (June 2018). “Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization”. en. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Salt Lake City, UT: IEEE, pp. 1082–10828. ISBN: 978-1-5386-6100-0. DOI: 10.1109/CVPRW.2018.00143. URL: https://ieeexplore.ieee.org/document/8575297/.

UCL (2023). *Physical activity ratio (PAR)*. [Online; accessed 4-October-2023]. URL: https://www.ucl.ac.uk/~ucbcdab/enbalance/definitions.htm#PAR.

Vinkers, Christiaan H. et al. (2013). “The effect of stress on core and peripheral body temperature in humans”. In: *Stress* 16.5. PMID: 23790072, pp. 520–530. DOI: 10.3109/10253890.2013.807243. eprint: https://doi.org/10.3109/10253890.2013.807243. URL: https://doi.org/10.3109/10253890.2013.807243.

Virtanen, Pauli et al. (2020). “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

Walitza, Susanne et al. (2011). “Obsessive-Compulsive Disorder in Children and Adolescents”. In: *Dtsch Arztebl International* 108.11, pp. 173–179. DOI: 10.3238/arztebl.2011.0173. eprint: https://www.aerzteblatt.de/pdf.asp?id=81323. URL: https://www.aerzteblatt.de/int/article.asp?id=81323.

Wang, Fei et al. (Sept. 2019). “Learning from simulation: An end-to-end deep-learning approach for computational ghost imaging”. en. In: *Optics Express* 27.18, p. 25560. ISSN: 1094-4087. DOI: 10.1364/OE.27.025560.

Wei, Jing et al. (Feb. 2018). “Transdermal Optical Imaging Reveal Basal Stress via Heart Rate Variability Analysis: A Novel Methodology Comparable to Electrocardiography”. In: *Frontiers in Psychology* 9. DOI: 10.3389/fpsyg.2018.00098.

Wikipedia (2023). *Average human height by country — Wikipedia, The Free Encyclopedia*. [Online; accessed 4-October-2023]. URL: https://en.wikipedia.org/w/index.php?title=Average_human_height_by_country&oldid=1178375879.

Wikipedia contributors (2023). *Clothing insulation — Wikipedia, The Free Encyclopedia*. [Online; accessed 4-October-2023]. URL: https://en.wikipedia.org/w/index.php?title=Clothing_insulation&oldid=1174546636.

Wongvorachan, Tarid, Surina He, and Okan Bulut (Jan. 2023). “A Comparison of Undersampling, Oversampling, and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining”. en. In: *Information* 14.1, p. 54. ISSN: 2078-2489. DOI: 10.3390/info14010054.

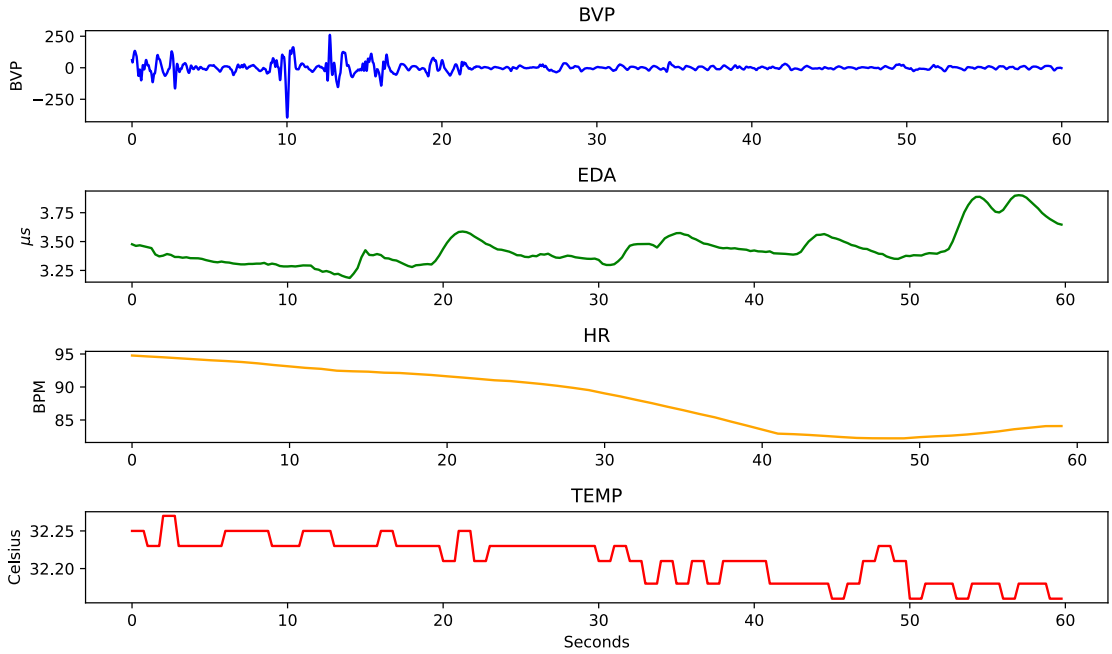
World, Disabled (2017). *Average Height to Weight Chart: Babies to Teenagers*. Accessed: 2023-10-04. URL: https://www.disabled-world.com/calculators-charts/height-weight-teens.php#referencedw.

Yang, Qiang and Xindong Wu (Dec. 2006). “10 CHALLENGING PROBLEMS IN DATA MINING RESEARCH”. en. In: *International Journal of Information Technology & Decision Making* 05.04, pp. 597–604. ISSN: 0219-6220, 1793-6845. DOI: 10.1142/S0219622006002258.

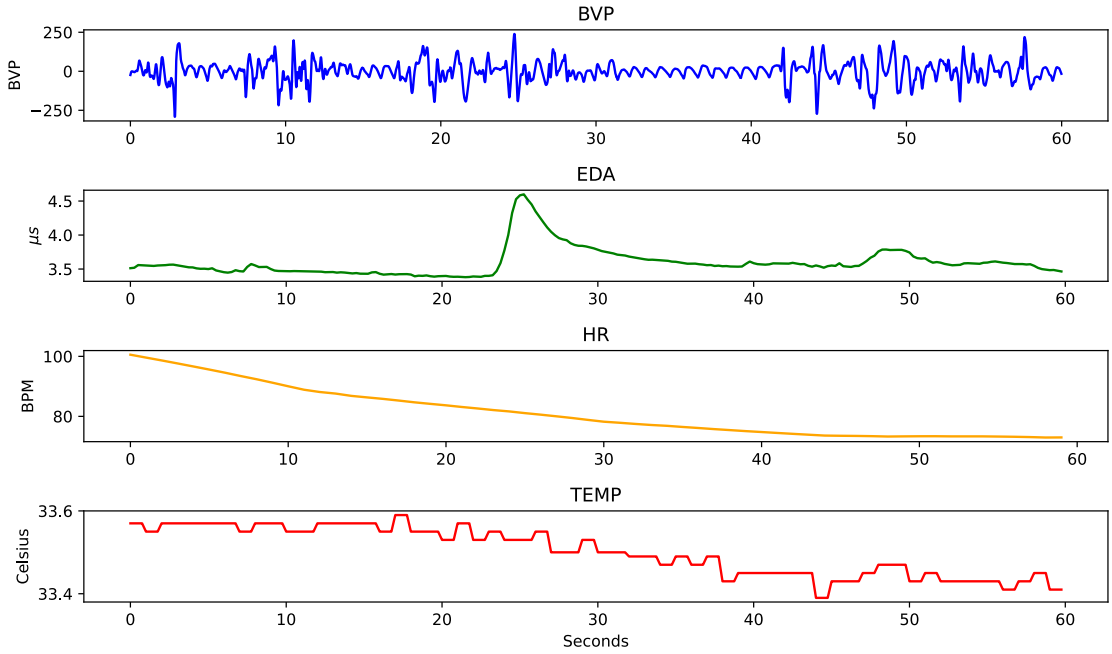
Zhao, Bendong et al. (2017). “Convolutional neural networks for time series classification”. In: *Journal of Systems Engineering and Electronics* 28.1, pp. 162–169. DOI: 10.21629/JSEE.2017.01.18.

A Appendix

A.1 Example of observations in DTU



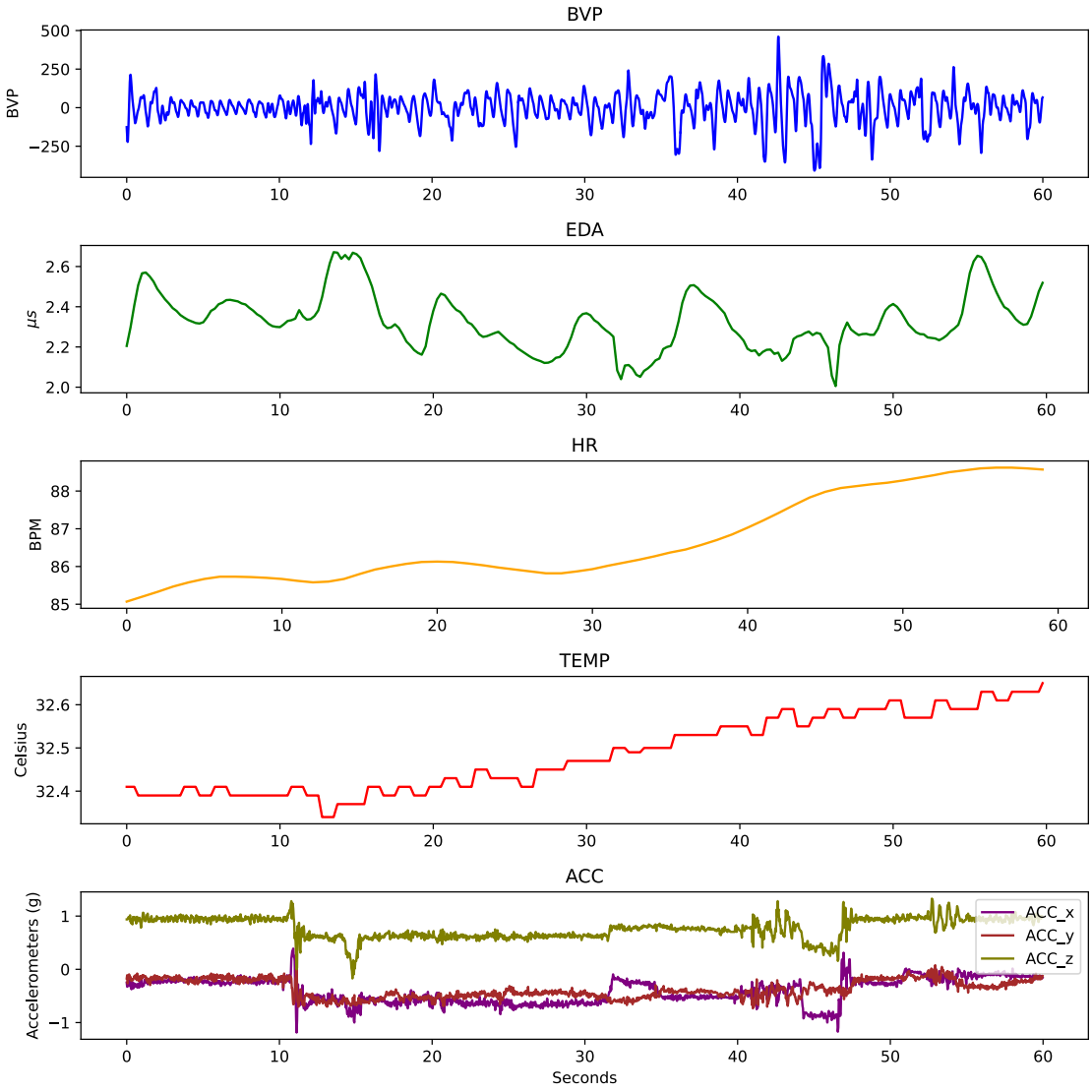
(a) An event observation.



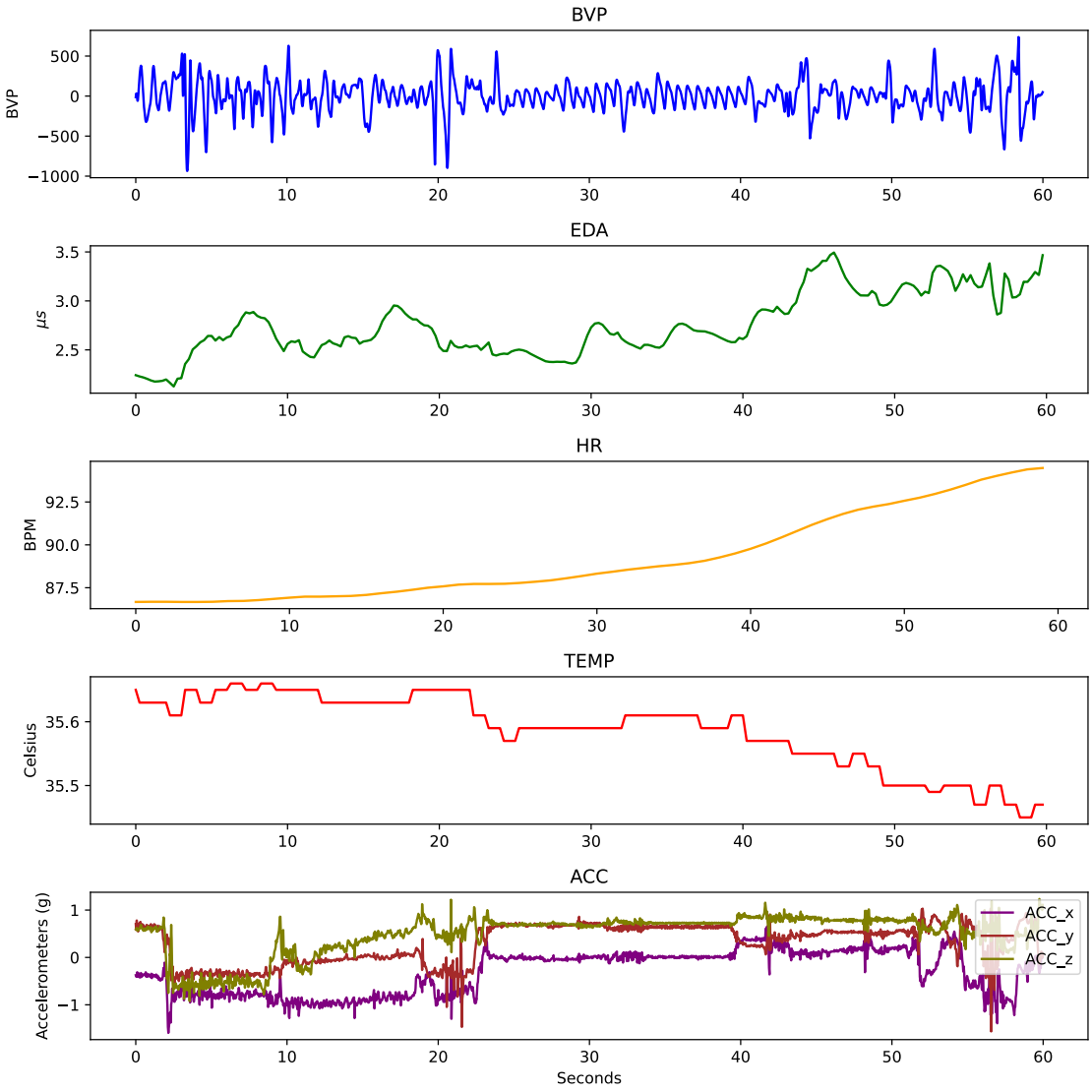
(b) A non-event observation.

Figure A.1: Participant 4 in the DTU dataset for an example of an event and non-event segment with a duration of 1 minute.

A.2 Example of observations in ROAD



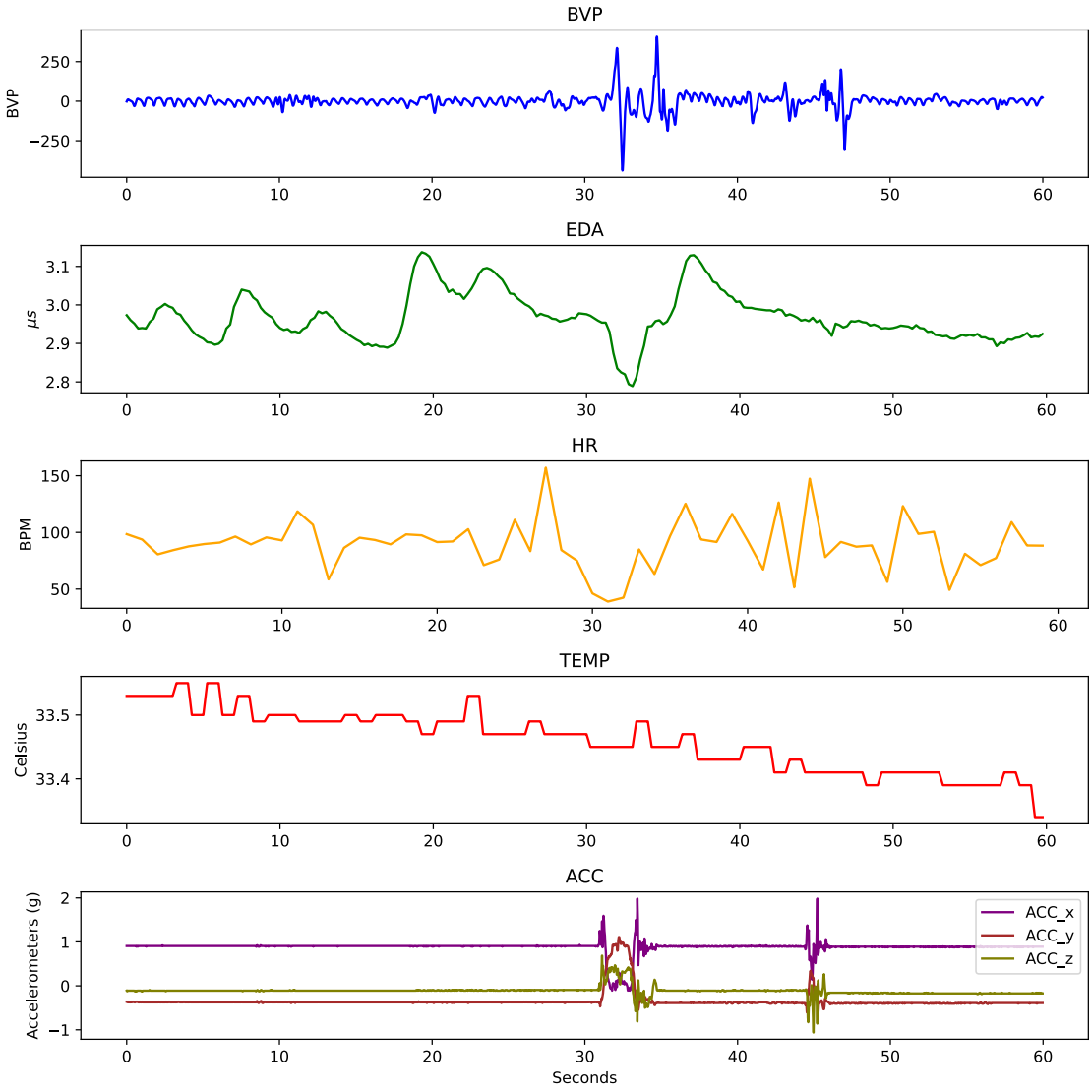
(a) An event observation.



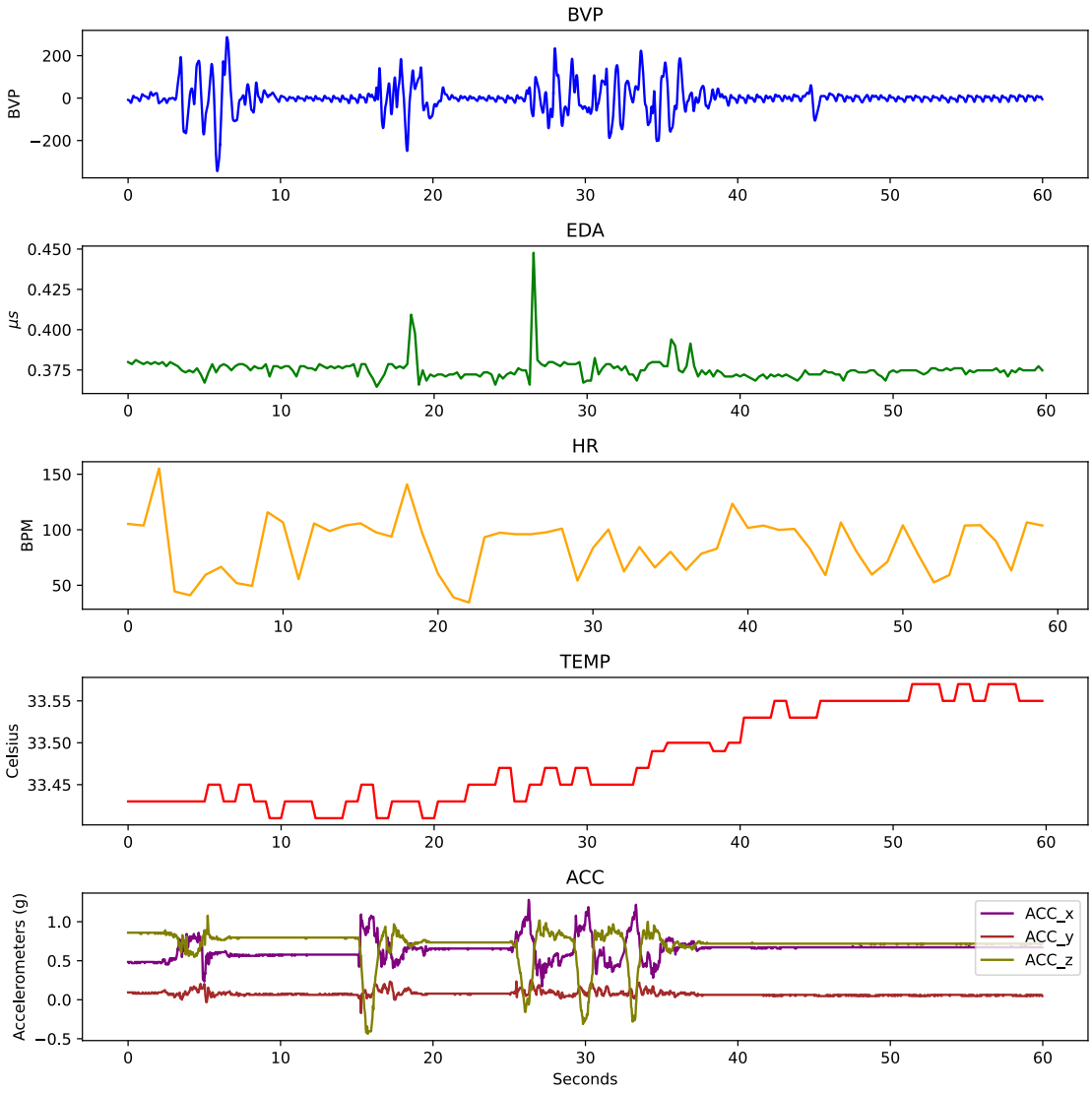
(b) A non-event observation.

Figure A.2: Participant 7 in the ROAD dataset for an example of an event and non-event segment with a duration of 1 minute.

A.3 Example of observations in WESAD



(a) An event observation.



(b) A non-event observation.

Figure A.3: Participant 0 in the WESAD dataset for an example of an event and non-event segment with a duration of 1 minute.

A.4 Histograms of signal-to-noise ratio for the different datasets.

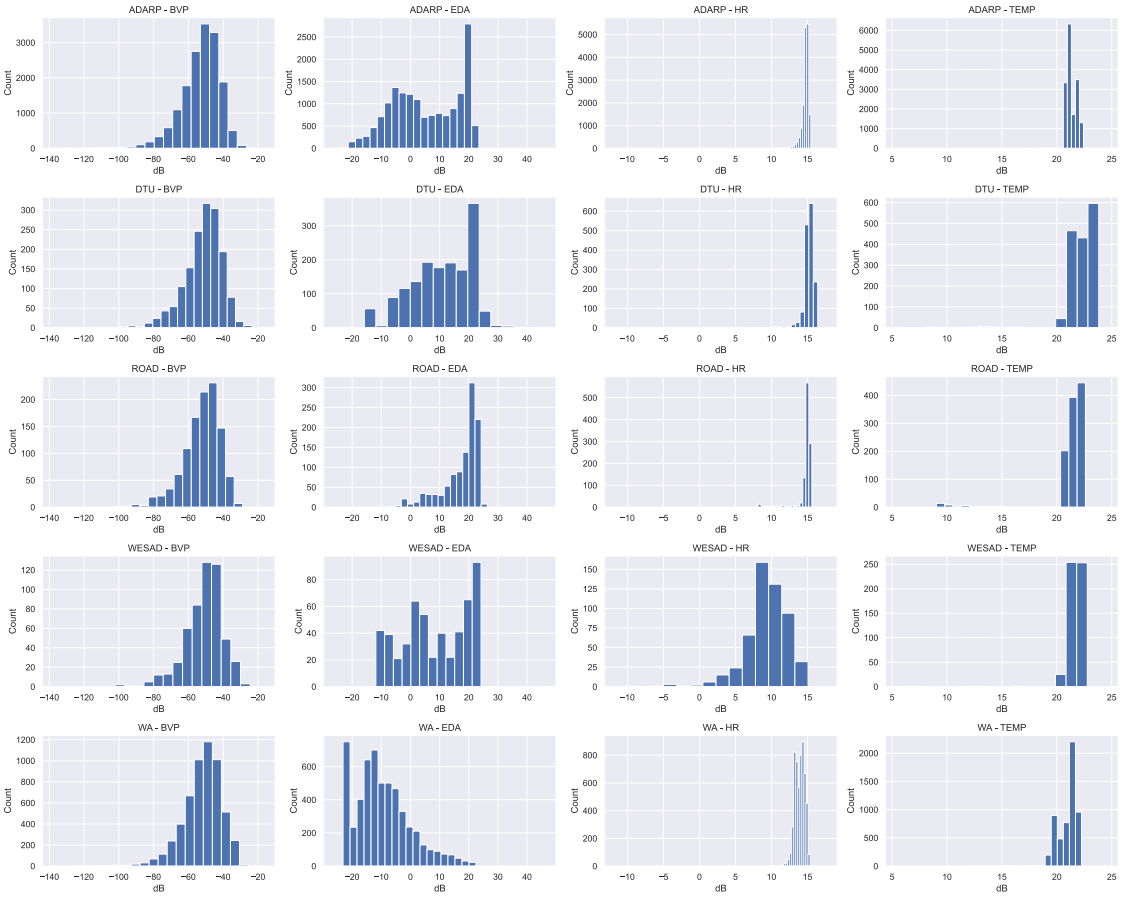


Figure A.4: Histograms of the observations' signal-to-noise ratio for each signal and each dataset. The columns are the signals, from left to right: BVP, EDA, HR, and TEMP. The rows are the datasets, from top to bottom: ADARP, DTU, ROAD, WESAD, and WA.

A.5 Histograms of mean for the different datasets. 102 WA TEMP outliers removed.

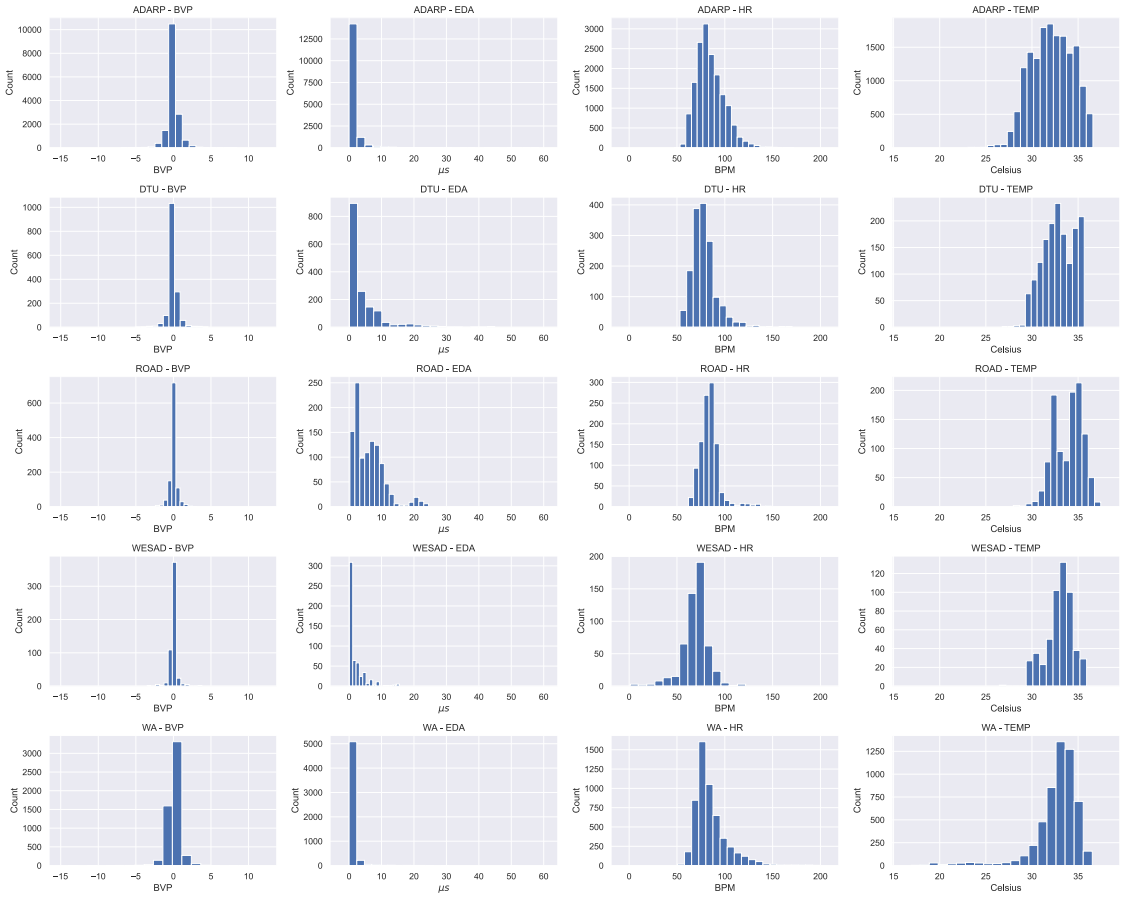


Figure A.5: Histograms of the observations' mean for each signal and each dataset. The columns are the signals, from left to right: BVP, EDA, HR, and TEMP. The rows are the datasets, from top to bottom: ADARP, DTU, ROAD, WESAD, and WA. The 102 outliers found in the WA dataset based on TEMP have been removed.

A.6 Violin plots for the different datasets for TEMP. 102 WA TEMP outliers removed.

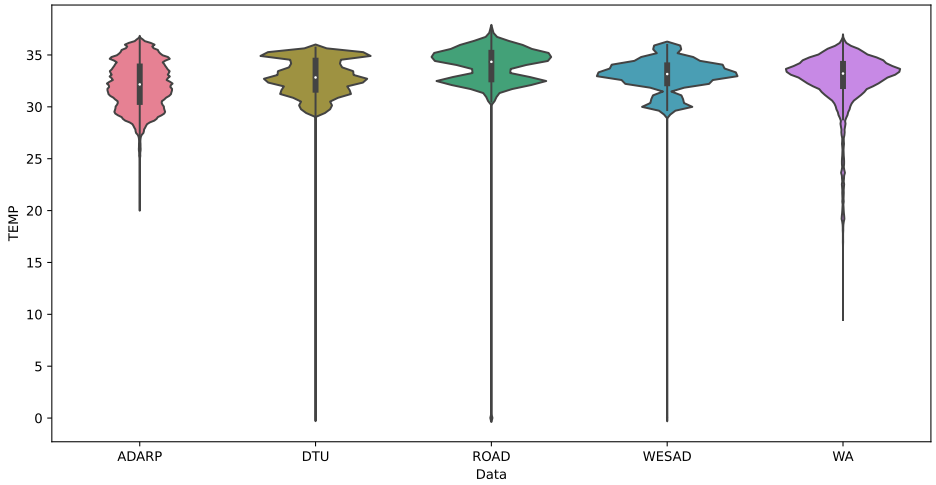


Figure A.6: Violin plot for TEMP for each dataset. The 102 outliers found in the WA dataset based on TEMP have been removed.

A.7 Violin plots for the different participants of WA for TEMP. 102 WA TEMP outliers removed.

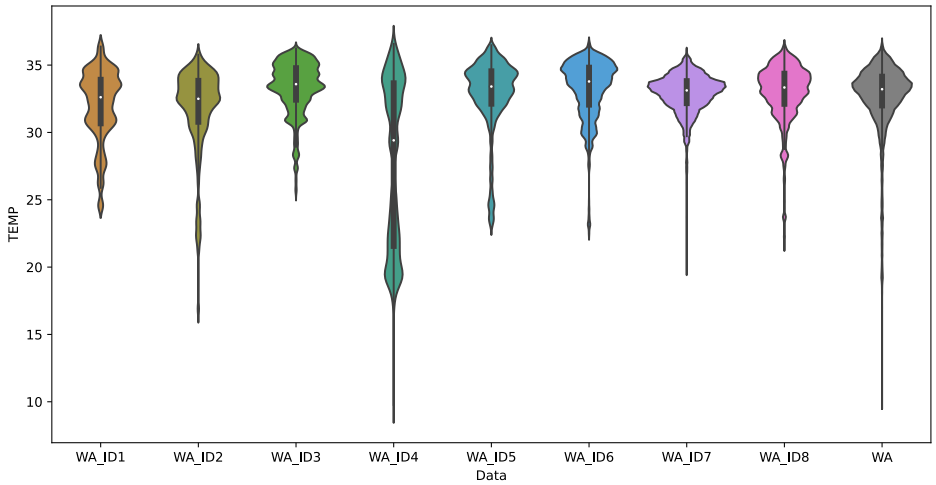
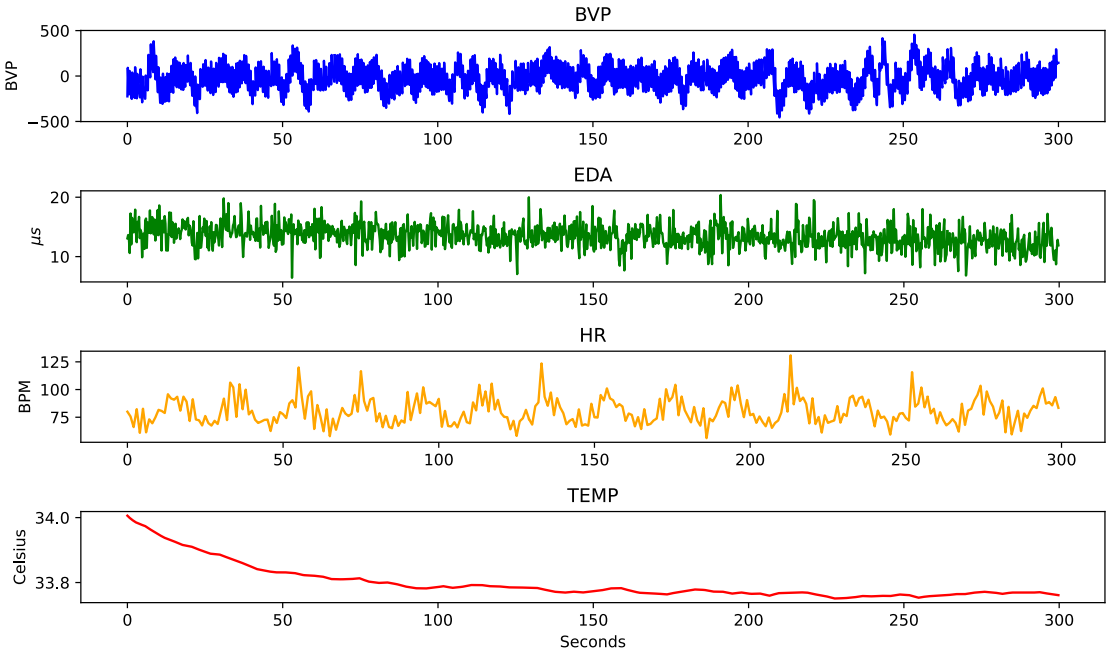
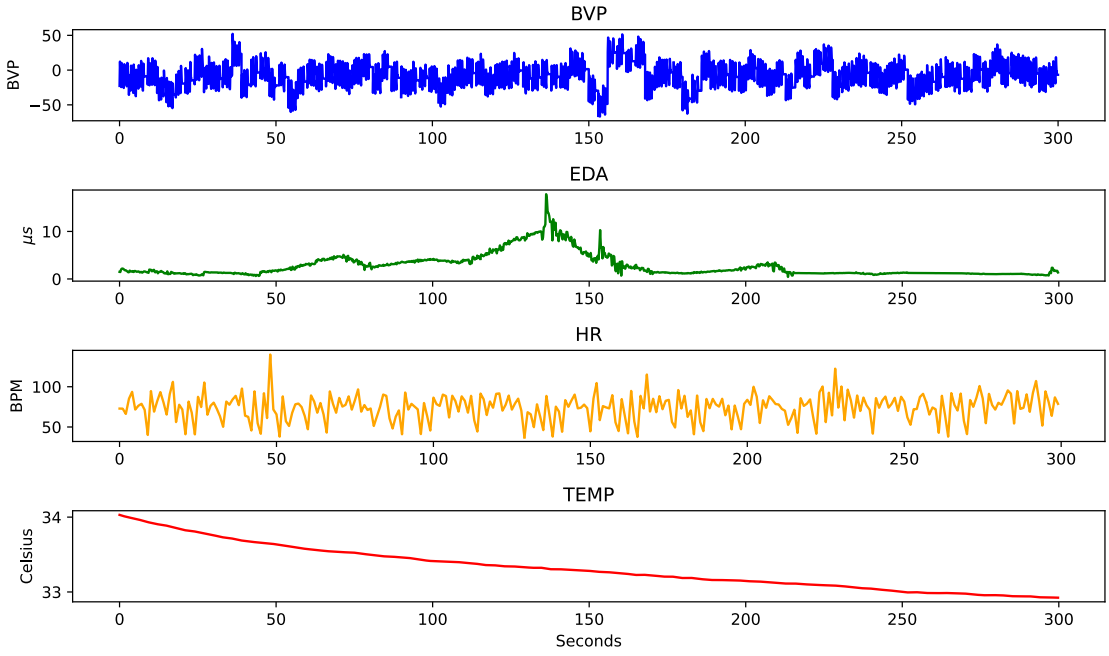


Figure A.7: Violin plot for TEMP for each participant. The 102 outliers found in the WA dataset based on TEMP have been removed.

A.8 Visulization of simulated data for 5 minutes



(a) The four simulated signals using the plain method.



(b) The four simulated signals using the fragmented method.

Figure A.8: The simulated data is for a 30-year-old female with normal physical condition in fall.

A.9 Explanation of input parameters' ranges for data simulation

	Parameter	Reasoning of selected range
BVP & HR	Average HR	Normal values for adults heart rate are between 60 and 100 according to Hipp 2023
	Standard deviation HR	Normal value around 8 Wei et al. 2018
	Physical condition effect on average HR	Athletes typically have a lower HR Hipp 2023
	Physical condition effect on standard deviation HR	Higher for better physical condition, lower for worse physical condition Boderskov 2023
	Genders effect on HR	Females typically have higher HR than males, around 6-12 bpm higher K et al. 2014
	The average HR for 5 to 9 years old	Five to nine years old typically have a HR between 70 and 115 Hipp 2023
	Scale value for BVP	Based on the Empatica's website Empatica 2023b
EDA	Drift for EDA	Based on visual inspection
	SCR peak	Based on visual inspection
	EDA amplitude of laplace noise	Based on visual inspection
TEMP	BMI	Normal weight BMI is between 18.5-25, underweight below, and overweight above Initiative 2000, but can vary more for athletes Matte 2019
	Weight	Calculated based on BMI and height CDC 2022
	Height for children	Estimated based on average height for children age groups World 2017
	Height for adults	Estimated based on average height around the world for males and females respectively Wikipedia 2023
	Outside humidity, temperature and windspeed	Estimated based on season min and max in Denmark from DMI DMI 2023
	Indoor humidity, temperature and windspeed	Estimated based on ideal indoor humidity and temperature with added variance Panasonic 2023
	Clothing insulation, I_{cl}	Personal estimations for different weather conditions based on Wikipedia contributors 2023
	CI based on physical condition	Cardiac index is typically between 2.5 and 4, where athletes have higher values Patel N 2022
	Posture	These are the available inputs from JOS-3
	Activity	Using physical activity ratios, which are not actual exercise UCL 2023

Table A.1: Reasons and references for the chosen ranges of input parameters for data simulation.

A.10 Visualization of different BVP filter

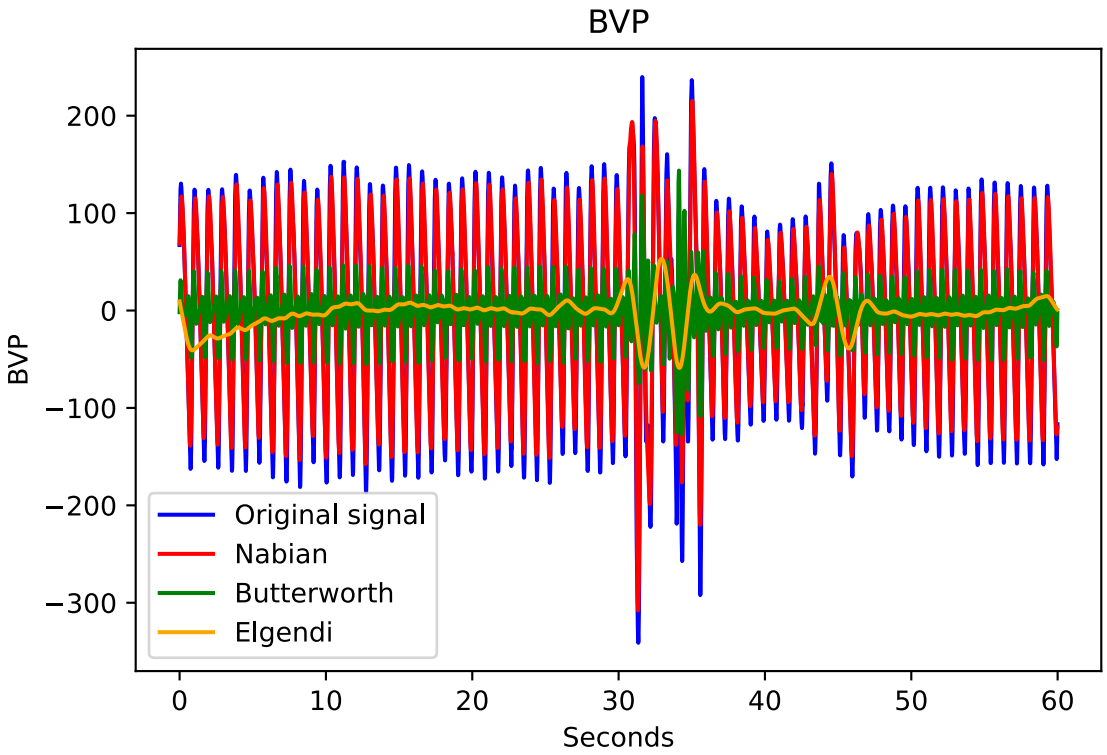


Figure A.9: A visualization of how a BVP signal is distorted after various filters. The blue line is the original signal from participant number 3 in the ADARP dataset during an event observation. The green line is using a 2. order Butterworth bandpass filter with cut-off frequencies of 2Hz and 12Hz using `scipy.signal.butter` function. The red and yellow lines are filtered using `Neurokit2`'s `ppg_clean` function which filters the signal using the specified method. By visual inspection, we see that the Nabian filter distorts the original signal the least. While the Butterworth filter significantly lowers the amplitude of the signal. Last the Elgendi filter has distorted the signal significantly so that the first 30 seconds almost look like a straight line.

A.11 Visulization of signals after standardization pr observation

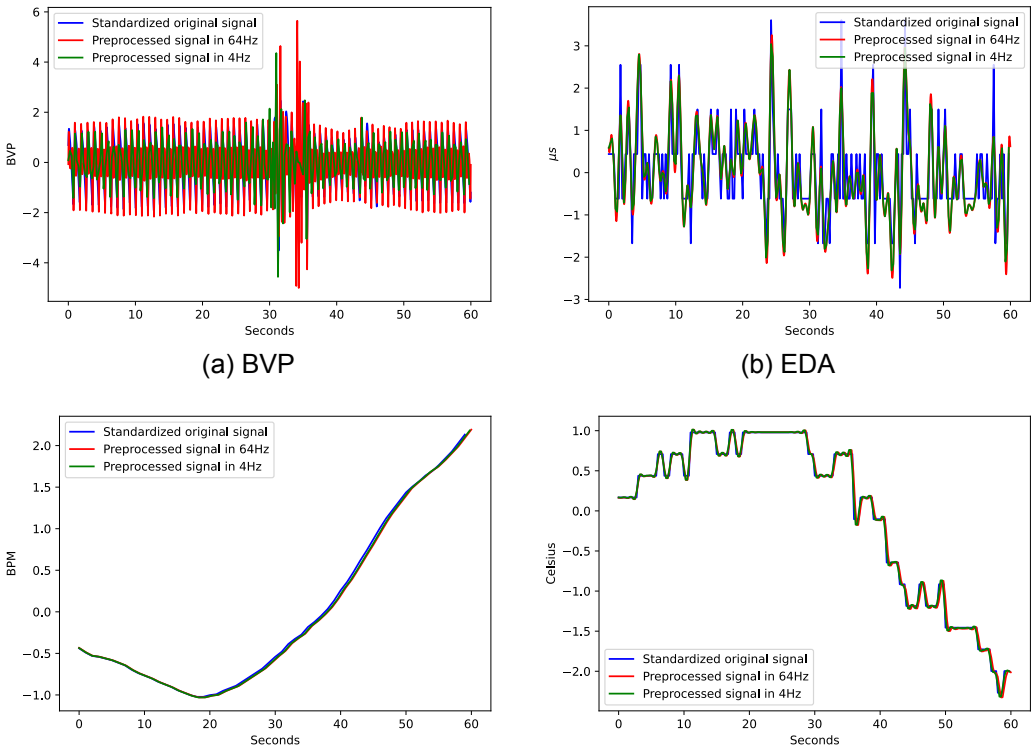
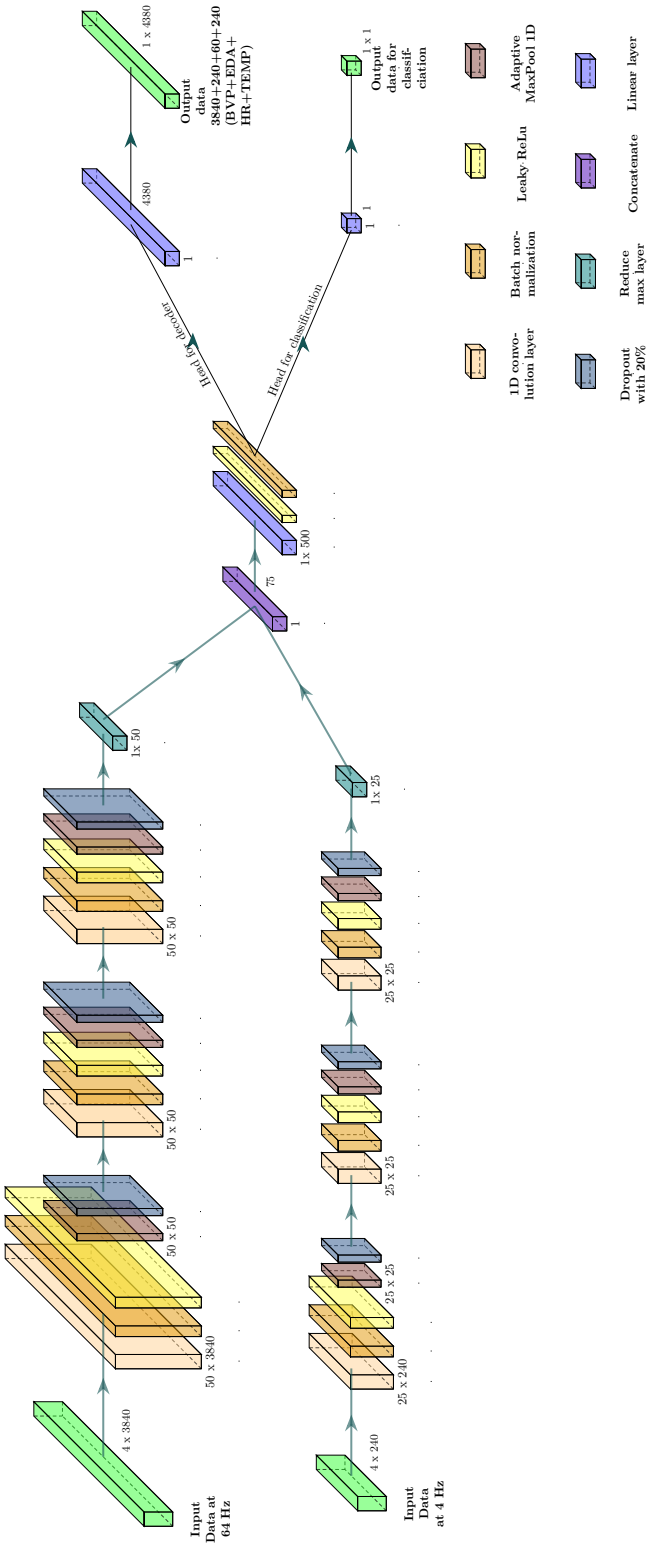


Figure A.10: An example of how an observation after standardizing the signals using the pr observation method. The blue line is the original signal, the red is the signal in 64Hz after pre-processing, and the green line is the signal in 4Hz after pre-processing.

A.12 Visualization of the model



A.13 Keras model overview

For each model, we tried out different architectures and did a bit of hyperparameter tuning. The displayed autoencoder models are the best outcomes for each model type. The classification for all models was a simple Dense network, with features from the encoder as input.

Dataset:	WESAD 1min		DTU 1min	
Model	Accuracy	F1	Accuracy	F1
LSTM	Train: 0.95 **Val: 0.93** **Test: 0.93**	Train: 0.91 **Val: 0.86** **Test: 0.81**	Train: 0.74 **Val: 0.72** **Test: 0.72**	Train: 0.51 **Val: 0.44** Test: 0.37
CNN	**Train: 1.00** Val: 0.91 Test: 0.91	**Train: 1.00** Val: 0.82 Test: 0.63	**Train: 0.92** Val: 0.66 Test: 0.66	**Train: 0.87** Val: 0.37 **Test: 0.52**
Dense	**Train: 1.00** Val: 0.91 Test: 0.91	**Train: 1.00** Val: 0.82 Test: 0.57	Train: 0.91 Val: 0.70 Test: 0.70	Train: 0.85 Val: 0.32 Test: 0.44

Table A.2: Overview of performance for selected models in Keras.

A.14 Further analysis of FCN models

FCN models	Accuracy	F1	Run time
Bottleneck: 4Hz 50, 64Hz 100	Train: 0.73 Val: 0.68 Test: 0.69	Train: 0.66 Val: 0.63 Test: 0.61	4m27s
Bottleneck: 4Hz 50, 64Hz 100 Recon: Extra linear layer from 150 ->500 ->length of signal	Train: 0.69 Val: 0.68 Test: 0.62	Train: 0.63 Val: 0.64 Test: 0.56	5m1s
Above model with continuing classification training	Train: 0.71 Val: 0.68 Test: 0.64	Train: 0.66 Val: 0.67 Test: 0.62	7m32s
Bottleneck: 4Hz 50, 64Hz 100 Recon: Extra linear layer from 150 ->500 ->length of signal	**Train: 0.81** **Val: 0.7** **Test: 0.75**	**Train: 0.76** **Val: 0.68** **Test: 0.7**	Not known
Bottleneck: 4Hz 50, 64Hz 200 Recon: Extra linear layer from 250 ->500 ->batch norm ->length of signal Class: Extra linear layer from 250 ->25 ->batch norm ->length of signal	Train: 0.71 Val: 0.6 Test: 0.59	Train: 0.68 Val: 0.61 Test: 0.59	19m33s
Bottleneck: 4Hz 24, 64Hz 50 Recon: Extra linear layer from 74 ->148 ->batch norm ->length of signal	Train: 0.75 Val: 0.68 Test: 0.7	Train: 0.7 Val: 0.64 Test: 0.63	**2m56s**

Table A.3: Accuracy and F1-scores of different FCN models. Each model has been run once on the DTU+WESAD 1-min dataset.

FCN models	Accuracy	F1
124 ->62 ->2	Train: 0.73 Val: 0.66 Test: 0.66	Train: 0.65 Val: 0.58 Test: 0.57
124 ->500 ->2	Train: 0.65 Val: 0.62 Test: 0.54	Train: 0.63 Val: 0.61 Test: 0.53
124 ->62 ->2 LReLU in last linear	Train: 0.69 Val: 0.62 Test: 0.63	Train: 0.64 Val: 0.59 Test: 0.6
124 ->62 ->2 LReLU in network +lr/10	**Train: 0.8** **Val: 0.73** **Test: 0.75**	**Train: 0.76** **Val: 0.7** **Test: 0.7**
124 ->62 ->2 PReLU in network	Train: 0.75 Val: 0.64 Test: 0.66	Train: 0.7 Val: 0.64 Test: 0.63
124 ->62 ->2 PReLU in network +lr/10	Train: 0.75 Val: 0.64 Test: 0.66	Train: 0.7 Val: 0.64 Test: 0.63

Table A.4: Accuracy and F1-score of different FCN model architectures. Each model has been run once on the DTU+WESAD 1-min dataset.

A.15 Overview of different FCN models tested

FCN Model	Accuracy	F1
124->124/2->2	Train: 0.87 **Val: 0.84** Test: 0.65	Train: 0.81 **Val: 0.78** Test: 0.65
5*h->2*h ->h	Train: 0.79 Val: 0.77 **Test: 0.71**	Train: 0.73 Val: 0.72 **Test: 0.69**
5*h ->h	Train: 0.82 Val: 0.71 **Test: 0.71**	Train: 0.76 Val: 0.64 **Test: 0.69**
2*h ->h	Train: 0.66 Val: 0.61 Test: 0.52	Train: 0.63 Val: 0.62 Test: 0.59
POOL	Train: 0.81 Val: 0.74 Test: 0.65	Train: 0.73 Val: 0.69 Test: 0.56
POOL+k[16, 5, 3]	Train: 0.81 Val: 0.71 Test: 0.58	Train: 0.76 Val: 0.67 Test: 0.55
POOL+k[3, 5, 3]	Train: 0.81 Val: 0.71 Test: 0.58	Train: 0.76 Val: 0.67 Test: 0.55
pool final + k[7,5,3]	Train: 0.71 Val: 0.71 Test: 0.61	Train: 0.68 Val: 0.69 Test: 0.65
pool final + c[int(h/2), h, 2*h]	Train: 0.72 Val: 0.68 Test: 0.61	Train: 0.68 Val: 0.64 Test: 0.62
pool final + c[int(h/2), h, 2*h]+ ->500->2	Train: 0.75 Val: 0.68 Test: 0.55	Train: 0.7 Val: 0.67 Test: 0.59
pool final + c[int(h/2), h, 2*h]+ ->500->2 only 64Hz 100 nodes	Train: 0.73 Val: 0.65 Test: 0.68	Train: 0.68 Val: 0.65 **Test: 0.69**
pool final + c[int(h/2), h, 2*h]+ ->500->2 only 4Hz 50 nodes	Train: 0.82 Val: 0.71 Test: 0.65	Train: 0.75 Val: 0.67 Test: 0.62
same as before 100 nodes 64Hz	Train: 0.65 Val: 0.65 Test: 0.52	Train: 0.62 Val: 0.65 Test: 0.59
4 Layers: 2*feats->h->2*h->h	Train: 0.67 Val: 0.77 Test: 0.61	Train: 0.62 Val: 0.72 Test: 0.62
6 layers [2*feasts,h, 2*h, 5*h, 2*h, h]	Train: 0.71 Val: 0.71 Test: 0.55	Train: 0.67 Val: 0.71 Test: 0.61
6 with [0, 0, 0, 0, 1, 2, 4, 4, 4, 6, 6, 50, 0, 0, 0, 10]	**Train: 0.91** Val: 0.74 Test: 0.61	**Train: 0.88** Val: 0.71 Test: 0.54
6 with [0, 0, 0, 0, 1, 2, 4, 4, 4, 6, 6, 50, 0, 0, 0, 10]	Train: 0.72 Val: 0.71 Test: 0.55	Train: 0.66 Val: 0.67 Test: 0.53
6 with [1, 1, 1, 1, 1, 2, 4, 4, 4, 6, 6, 20, 1, 1, 1, 10]	Train: 0.68 Val: 0.68 Test: 0.55	Train: 0.64 Val: 0.67 Test: 0.59
50 4Hz, 25 64Hz	Train: 0.85 Val: 0.81 Test: 0.58	Train: 0.78 Val: 0.73 Test: 0.55

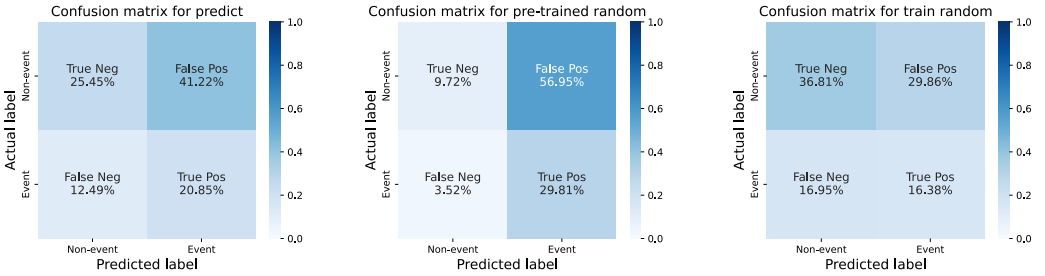
Table A.5: Test of different FCN architectures. All models were run on the DTU+WESAD 5-min dataset.

A.16 Test of Butterworth filter for BVP on the ADARP dataset

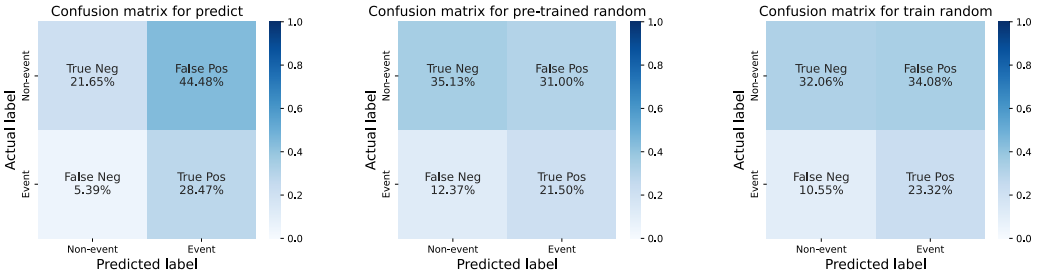
Butterworth filter			F1-score	
Highcut	Lowcut	Order	μ	σ
No filter (I assume)			0.49	0.08
8	0.5	6	0.63	0.04
12	0.5	6	0.63	0.05
12	2	6	0.64	0.04
12	2	5	0.59	0.04
12	2	2	0.65	0.04
12	4	2	0.64	0.05
16	4	2	0.63	0.05
12	2	1	0.65	0.05
14	2	2	**0.66**	**0.02**
6	2	2	0.54	0.10

Table A.6: Average and standard deviation F1-score after fine-tuning on ADARP as the test dataset. Each filter was run 10 times.

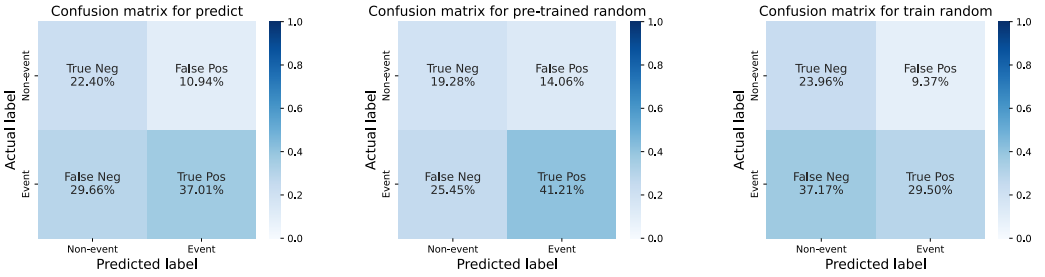
A.17 Confusion matrix for open-source dataet



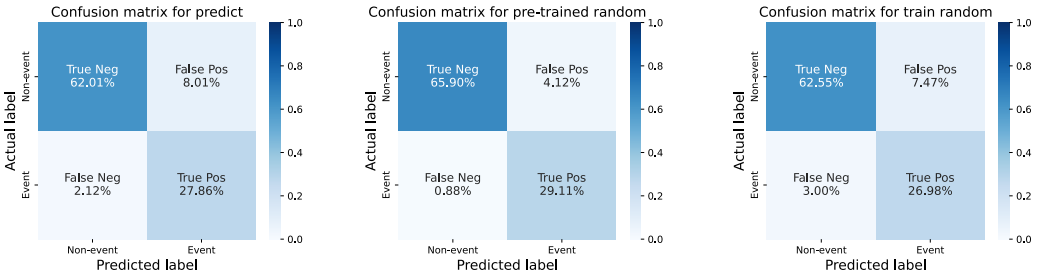
(a) ADARP predict (b) ADARP pre-train random (c) ADARP train random



(d) DTU predict (e) DTU pre-trained random (f) DTU train random



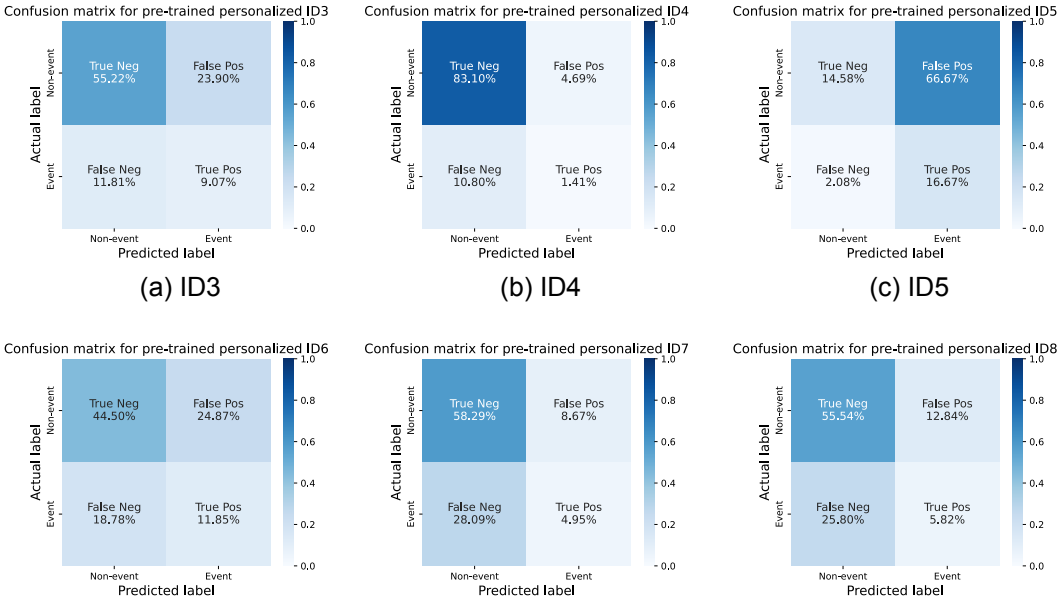
(g) ROAD Predict (h) ROAD pre-trained random (i) ROAD train random



(j) WESAD predict (k) WESAD pre-trained random (l) WESAD train random

Figure A.11: Confusion matrix for three different scenarios with 1-minute window length, 0-minute lead prediction time, and without activity classification for all open-source datasets. The results are reported in percentage and is averaged across 10 runs and contain all 3 splits: training, validation, and test sets.

A.18 Confusion matrix for each participant in pre-trained personalzied



(a) ID3 (b) ID4 (c) ID5

(d) ID6 (e) ID7 (f) ID8

Figure A.12: Confusion matrix for each participant in the scenario pre-train personalized with 1-minute window length, 0-minute lead prediction time, and without activity classification for all open-source datasets. The results are reported in percentage and are averaged across 10 runs and contain all 3 splits: training, validation, and test sets.

A.19 Statistical Analysis Plan (SAP)

APPLYING PRE-TRAINED DEEP-LEARNING MODEL ON WRIST ANGEL DATA - AN ANALYSIS PLAN

A PREPRINT

Harald Vilhelm Skat-Rørdam [*] [†]
s175393@dtu.dk

Mia Hang Knudsen [*] [†]
s183998@dtu.dk

Simon Nørby Knudsen [*] [†]
s174479@dtu.dk

Nicole Nadine Lønfeldt [‡]
nicole.nadine.loenfeldt@regionh.dk

Sneha Das [†]
sned@dtu.dk

Line Katrine Harder Clemmensen [†]
lkhc@dtu.dk

December 15, 2023

We aim to investigate if we can improve predictions of stress caused by OCD symptoms using pre-trained models, and present our statistical analysis plan in this paper. With the methods presented in this plan, we aim to avoid bias from data knowledge and thereby strengthen our hypotheses and findings.

The Wrist Angel study, which this statistical analysis plan concerns, contains data from nine participants, between 8 and 17 years old, diagnosed with obsessive-compulsive disorder (OCD). The data was obtained by an Empatica E4 wristband, which the participants wore during waking hours for 8 weeks. The purpose of the study is to assess the feasibility of predicting the in-the-wild OCD events captured during this period.

In our analysis, we aim to investigate if we can improve predictions of stress caused by OCD symptoms, and to do this we have created a pre-trained model, trained on four open-source data for stress prediction. We intend to apply this pre-trained model to the Wrist Angel data by fine-tuning, thereby utilizing transfer learning. The pre-trained model is a convolutional neural network that uses blood volume pulse, heart rate, electrodermal activity, and skin temperature as time series windows to predict OCD events. Furthermore, using accelerometer data, another model filters physical activity to further improve performance, given that physical activity is physiologically similar to stress.

By evaluating various ways of applying our model (fine-tuned, non-fine-tuned, pre-trained, non-pre-trained, and with or without activity classification), we contextualize the problem such that it can be assessed if transfer learning is a viable strategy in this domain.

Keywords Deep Learning · OCD · Predict · Obsessive-Compulsive Disorder · Children · Teens · Adolescents · AI · Artificial Intelligence · Mental Health · Empatica E4 · Transfer Learning

[*]These authors contributed equally

[†]Applied Mathematics and Computer Science, Technical University of Denmark, Kongens Lyngby, Denmark

[‡]Child and Adolescent Mental Health Center, Copenhagen University Hospital, Mental Health Services Copenhagen, Hellerup, Denmark

1 Scope of analysis

This analysis aims to evaluate whether transfer learning is a viable option for enhancing OCD-event predictability, which could be generalized to other areas of interest. This is especially relevant for new prediction tasks, where data scarcity is problematic.

This is done by assessing if a pre-trained deep-learning model can improve the performance of predicting obsessive-compulsive disorder (OCD) events compared to traditional machine learning techniques that were previously applied Lønfeldt et al. [2023], using the same data set, from here on referred to as Wrist Angel data.
A proposed daily plan for data analysis on Wrist Angel data can be seen in section 7.2.

The primary questions this analysis aims to answer are:

1. To what extent does deep learning improve performance compared to traditional machine learning techniques?
2. How might the performance of a pre-trained model (with and without fine-tuning) compare to that of a non-pre-trained model?
3. What is the influence of using a simple activity model on stress prediction?
4. How does predicting OCD events at various time intervals, such as 0, 1, 2, 3, 4, and 5 minutes before the event, affect performance? This interval between prediction and event will be referred to as prediction lead time.
5. How does the length of the signal sequence influence model performance?

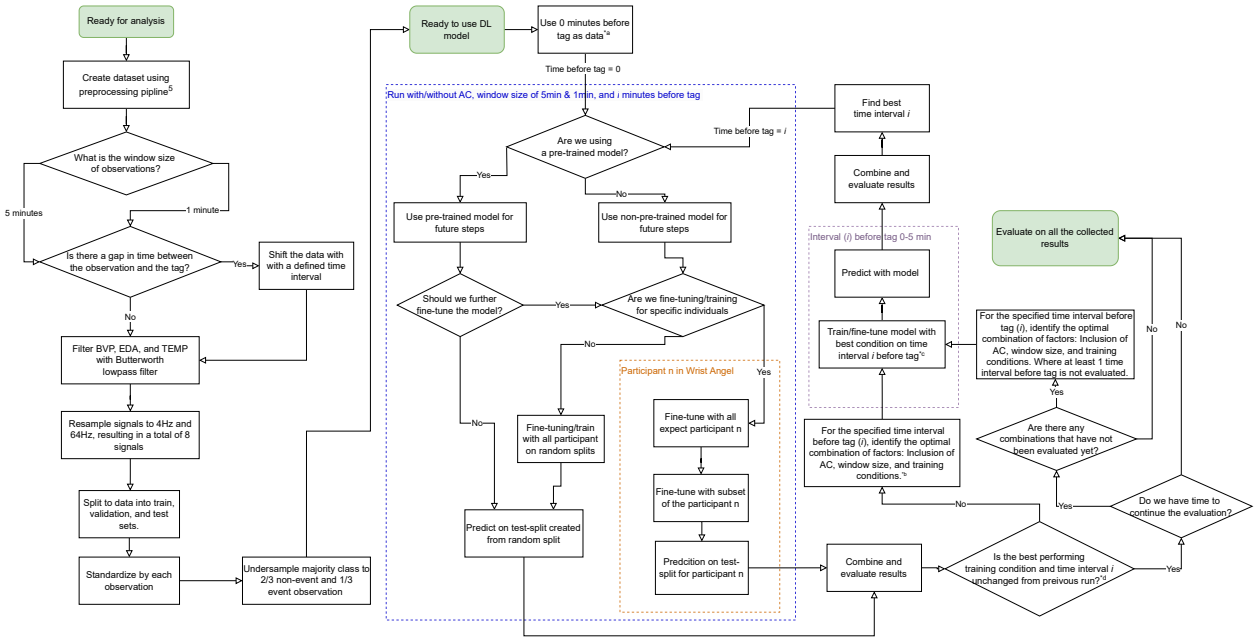


Figure 1: A flow chart describing the steps in signal processing and data analysis. The performance of models will be logged in table 1. *(All figures are PDF).*
Abbreviations. BVP: Blood volume pulse, EDA: Electrodermal activity, HR: Heart rate, TEMP: Skin temperature, DL: Deep learning, AC: Activity classification.
* For a visual example of the optimization steps see table 2 in Appendix. The letters correspond to the sub-tables.
[5] The preprocessing pipeline is the same as used in Lønfeldt et al. [2023].

2 Proposed Data Preprocessing

To get an understanding of the data, we will perform an exploratory analysis, computing metrics such as means and standard deviations for continuous variables and frequencies for categorical variables. Furthermore, we will plot histograms for physiological and accelerometer data. The accelerometer data will be used to evaluate and tune the threshold for a physical activity classification.

The preprocessing pipeline will be the same one used in the previous study and is explained in Lønfeldt et al. [2023] with the possibility of minor changes as explained below.

The data used in this study comes from the E4 wristband, 4 signals will be used in our model: Blood volume pulse (BVP), heart rate (HR), electrodermal activity (EDA), and skin temperature (TEMP). The tags and the timestamp related to the tag are used to pinpoint the stress episodes and to create the dataset. Additionally, accelerometer data is used for our physical activity classification.

We will extract signals using the window strategy and test different window sizes of 60 seconds and 5 minutes. There will be a 5-minute buffer after the tag where no data will be extracted. There will be no overlapping signals.

Experiments will be done with different prediction lead times to see how early we can predict an OCD event. Specifically, prediction lead times of 0, 1, 2, 3, 4, and 5 minutes. These are practically gaps between the end of the signal and the OCD event tag. It is done by extracting signals ending at the same time as the tag and increasing the interval up to 5 minutes with a 1-minute interval, resulting in a total of the above-mentioned 6 different ways of extracting event signals before the tag.

A sixth-order Butterworth low-pass filter with a cut-off frequency of 1Hz will be applied to EDA and TEMP. For BVP a second-order Butterworth filter with a high cut-off frequency of 12 Hz and a low cut-off frequency of 2 Hz will be used, while HR will remain unprocessed.

We plan to re-sample the data to obtain the same number of samples for all signals. Both down-sampled (4Hz) and up-sampled (64Hz) data will be used. BVP and EDA will be re-sampled using python `scipy.signal` which uses a Fourier method. HR and TEMP are re-sampled using linear interpolation.

The final step shall be splitting the data and standardizing the signals. This will be done by first splitting the data into train, validation, and test sets. The selected standardization method is observation-wise, meaning, we will standardize the data one observation at a time, each signal individually in all three splits.

To deal with the unbalanced data, under-sampling is used, by randomly selecting non-event windows from all non-event data. The final distribution of our data will be 1/3 event data and 2/3 non-event data.

3 Data sets used to pre-train our model

We have pre-trained our deep learning model on four different open-source data sets. The datasets used are:

1. The DTU dataset
   It is collected through an experiment conducted at DTU. A total of 28 participants spread over three cohorts participated in the experiment. All participants wore an Empatica E4 wristband. The experiment consisted of three 5-minute phases. First, a pre-task resting phase, then, the task-solving phase, and finally, a post-task resting phase. The timed puzzle task was categorized as a stress event.
2. WESAD Schmidt et al. [2018]
   WESAD is a dataset combining physiological data with stress exposure. This dataset contains data from 15 participants, each wearing an Empatica E4. The participants went through a stressful scenario, where they were exposed to the Trier Social Stress Test (TSST)Kirschbaum et al. [1993]. The stress test consists of a 5-minute speech and a mental arithmetic task. The non-stressful category comes from scenarios of resting at a table and watching funny video clips.
3. AffectiveROAD dataset Haouij et al. [2018]
   This is from an experiment with 14 drives, from 10 different participants wearing two Empatica E4s. All participants drive the same route for three different types of roads, being exposed to stressful and non-stressful conditions. The "stress" metric is created based on a subjective measure from a co-driver, and the driver itself.
4. ADARP dataset Sah et al. [2022]
   A dataset was created by Washington State University, to examine if it is possible to detect stress in subjects diagnosed with alcohol use disorder (AUD). The data was collected from 11 participants over an average of 7 days, where they were asked to wear an E4 wristband during their daily life and tag times when they felt stressed.

4 Deep Learning Modeling

The event classification model we wish to apply to the Data from the Wrist Angel Feasibility Study consists of a 1D convolutional auto-encoder, from which we use the encoder and attach it to a convolutional classification network.

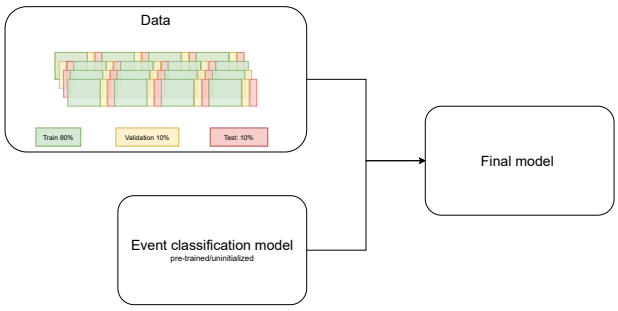
Details about the model can be seen in section 7.1. This model will be used both as a pre-trained model and as a non-pre-trained model.

Furthermore, we intend to include a threshold for standard deviation or Fourier transform frequency to classify physical activity based on accelerometer data, before using the event classification model. To determine which of the two classification methods, standard deviation or Fourier transform frequency, will be used, the in-clinic session with dance and relax data is used to create a baseline for when the children are in physical activity. The method that can most effectively distinguish the activity will then be used as our activity classification.

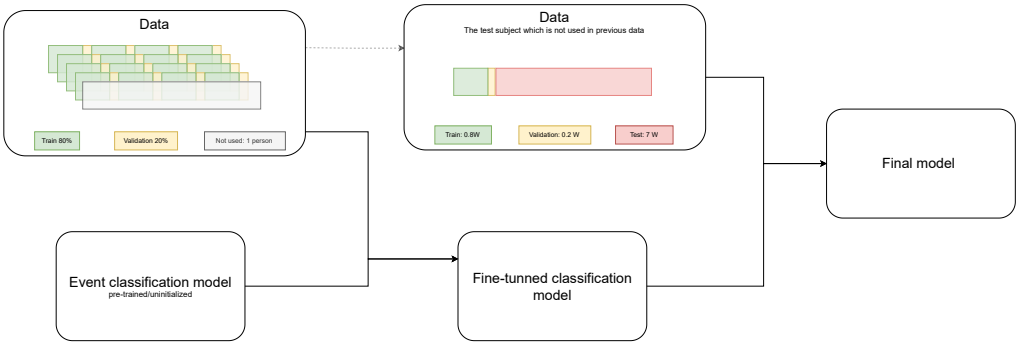
When using the physical activity classification, all observations classified as physical activity will be removed from the data, before performing any fine-tuning/predicting. If 25% (based on statistics from Mota et al.) or more of the data is removed due to the physical activity classification, we will re-sample the data to fine-tune and predict data with less physical activity. When the physical activity classification is not used to filter the data, no observations are removed, but will still be classified, such that we can compare the physical activity classification to the final OCD event classification at a later time.

We intend to apply our event classification model in five different ways:

1. Apply our pre-trained model directly to the Wrist Angel data to predict OCD events and assess the performance of the model, without the model knowing anything about the specific dataset.
2. Fine-tune our pre-trained model to a subset of the Wrist Angel data, and predict OCD events of the remaining data. Assess performance, to see if fine-tuning helps. We will fine-tune in two different ways:
   (a) Random (fig. 2a): Randomly splitting the Wrist Angel data in 80% train, 10% validation and 10% test.
   (b) Random + Personalized (fig. 2b): Two fine-tunes. First, we extract one person as the test subject. The remaining 8 persons will be randomly split into 80% train and 20% validation data for the first fine-tuning. Thereafter, a second fine-tuning will be applied to the test subject, where the first week will be split into 80% training and 20% validation. The remaining 7 weeks will be used for testing. This will then be repeated such that each person has been extracted, giving us a total of 9 tests.
3. Not using our pre-trained model, but training with the Wrist Angel data on our uninitialized model (not pre-trained). This will be used to assess if a pre-trained model improves performance. Training/fine-tuning in the same two ways as explained above with the pre-trained model.



(a) Flow for fine-tuning method: Random.



(b) Flow for fine-tuning method: Random + Personalized.

Figure 2: Visualization of the two fine-tuning/training methods.

We will apply our event classification model in the stated five different ways, both with and without doing a physical activity classification beforehand. This classification aims at removing time-series windows where the participant has performed physical activity, to not risk predicting it as an OCD event, since physical activity can have similar effects on physiological signals as stress.

5 Evaluation methods

To evaluate our models we will use the metrics accuracy and F1-score.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

$$\text{F1-score} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{2}$$

Where TP is true positives, TN is true negatives, FP is false positives, and FN is false negatives. The $F1\text{-}score$ is the harmonic mean of the two metrics $Precision$ and $Recall$.

Furthermore, we will plot Receiver Operating Characteristic (ROC) curves to investigate different thresholds for the classifications, and evaluate these using the Area Under the Curve (AUC).

We will evaluate our model for each of the 5 different ways as explained in section 4. Additionally, we will evaluate each of these with and without the physical activity classification. Furthermore, we will do everything for 60-second intervals and 5-minute intervals. Finally, we will do everything on the 6 different extracted signals, namely the different periods between the signal and OCD event tag. In total, this gives us 120 runs, as can be seen in table 1. We might not have time to evaluate our model in all these 120 ways. Hence, we intend to optimize the order by first evaluating everything for 0 min prediction lead time, then continue with the additional prediction lead times for the best result, etc. This strategy is visualized in fig. 1

$$\text{Number of runs} = 5 \cdot 2 \cdot 2 \cdot 6 = 120 \tag{3}$$

5min data Physical activity classification	Prediction lead time					
	0min	1min	2min	3min	4min	5min
Pre-trained - Predict directly (non fine-tuned)						
Pre-trained - Random fine-tuned						
Pre-trained - Random + personalized fine-tuned						
Uninitialized - Random train						
Uninitialized - Random + personalized train						
5min data No physical activity classification	Prediction lead time					
	0min	1min	2min	3min	4min	5min
Pre-trained - Predict directly (non fine-tuned)						
Pre-trained - Random fine-tuned						
Pre-trained - Random + personalized fine-tuned						
Uninitialized - Random train						
Uninitialized - Random + personalized train						
1min data Physical activity classification	Prediction lead time					
	0min	1min	2min	3min	4min	5min
Pre-trained - Predict directly (non fine-tuned)						
Pre-trained - Random fine-tuned						
Pre-trained - Random + personalized fine-tuned						
Uninitialized - Random train						
Uninitialized - Random + personalized train						
1min data No physical activity classification	Prediction lead time					
	0min	1min	2min	3min	4min	5min
Pre-trained - Predict directly (non fine-tuned)						
Pre-trained - Random fine-tuned						
Pre-trained - Random + personalized fine-tuned						
Uninitialized - Random train						
Uninitialized - Random + personalized train						

Table 1: The maximum of 120 ways we intend to apply our model to the Wrist Angel data. Depending on time restrictions we might not complete everything. We intend to first complete the 0 min column, then continue with the best result based on the F1-score, and complete this row for different prediction lead times. Thereafter, we will complete the full column for the best prediction lead time. Continuing interchanging between completing columns and rows, until everything is complete, or we run out of time.

6 Discussion

Results from our deep learning model will be compared to results from the traditional machine learning model presented in Lønfeldt et al. [2023]. Our model will be tested on the Wrist Angel data both as pre-trained with and without fine-tuning and as non-pre-trained, to examine if pre-training and fine-tuning improves performance. The model will be evaluated with and without removing the observations classified as physical activity to investigate if a simple physical activity classifier helps in stress prediction. The signals will be shifted with different gaps between the end of the signal and the OCD event tag, such that we can explore the performance of predicting OCD events at different times ahead. The Wrist Angel data is being split in windows of both 1min and 5min, to examine if the length of the time series influences the performance.

By publishing a statistical analysis plan before handling the Wrist Angel data, we ensure that the model and data parameters are set, such that, no data interpretation bias influences our results and decisions.

The Wrist Angel dataset introduces some limitations. First, the limited number of participants, and thus the generalizability of the model can not be properly evaluated. Second, given the in-the-wild nature of the data, a significant portion will most likely be influenced by noise, for instance, because the wristband is not properly worn.

References

Nicole Nadine Lønfeldt, Kristoffer Vinther Olesen, Sneha Das, Cecilie Mora-Jensen, Anne Katrine Pagsberg, and Line Katrine Harder. Predicting obsessive-compulsive disorder episodes in adolescents using a wearable biosensor - a wrist angel feasibility study. 2023.

Philip Schmidt, Attila Reiss, Robert Duerichen, Claus Marberger, and Kristof Van Laerhoven. Introducing wesad, a multimodal dataset for wearable stress and affect detection. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, ICMI '18, page 400–408, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356923. doi:10.1145/3242969.3242985. URL https://doi.org/10.1145/3242969.3242985.

Clemens Kirschbaum, Karl-Martin Pirke, and Dirk Hellhammer. The 'trier social stress test' – a tool for investigating psychobiological stress responses in a laboratory setting. *Neuropsychobiology*, 28:76–81, 02 1993. doi:10.1159/000119004.

Neska El Haouij, Jean-Michel Poggi, Sylvie Sevestre-Ghalila, Raja Ghozi, and Mériem Jaïdane. Affectiveroad system and database to assess driver's attention. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, SAC '18, page 800–803, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450351911. doi:10.1145/3167132.3167395. URL https://doi.org/10.1145/3167132.3167395.

Ramesh Kumar Sah, Michael McDonell, Patricia Pendry, Sara Parent, Hassan Ghasemzadeh, and Michael J. Cleveland. Adarp: A multi modal dataset for stress and alcohol relapse quantification in real life setting. (arXiv:2206.14568), June 2022. URL http://arxiv.org/abs/2206.14568. arXiv:2206.14568 [cs, eess].

Jorge Mota, Paula Santos, Sandra Guerra, José C. Ribeiro, and José A. Duarte. Patterns of daily physical activity during school days in children and adolescents. 15(4):547–553. ISSN 1520-6300. doi:10.1002/ajhb.10163. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/ajhb.10163. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ajhb.10163.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. *CoRR*, abs/1810.03993, 2018. URL http://arxiv.org/abs/1810.03993.

Iman Deznabi and Madalina Fiterau. Multiwave: Multiresolution deep architectures through wavelet decomposition for multivariate time series prediction, 2023.

Kristoffer Vinther Olesen, Nicole Nadine Lønfeldt, Sneha Das, Anne Katrine Pagsberg, and Line Katrine Harder Clemmensen. Predicting obsessive-compulsive disorder events in children and adolescents in the wild using a wearable biosensor (wrist angel): Protocol for the analysis plan of a nonrandomized pilot study. *JMIR Res Protoc*, 12: e48571, Nov 2023. ISSN 1929-0748. doi:10.2196/48571. URL https://www.researchprotocols.org/2023/1/e48571.

7 Appendix

7.1 Model Card Mitchell et al. [2018]

Model Card - CNN Stress Classification

Model Details
- Developed by Harald, Mia, and Simon at DTU, 2023 - Inspired by Deznabi and Fiterau [2023].
- Convolutional Neural Network.
- Pre-trained for stress classification. Will be fine-tuned for obsessive-compulsive disorder OCD event classification in children and adolescents.

Intended Use
- Intended use is to predict OCD events.
- Intended users are children with OCD, wearing an Empatica E4 wristband, such that the model can predict and warn, before an OCD event.
- Not specifically pre-trained for either children or OCD events, hence could be used for several age groups and various acute stress episodes.

Factors
- Relevant factors that might affect the model performance are personal factors such as a person's age, health, shape, and environmental factors such as temperature, and humidity.
- Evaluation factors are 1: with or without physical activity classification, 2: using respectively 1min and 5min intervals, 3: varying the gap between the signal and the event tag (0 to 5min), and 4: the five different ways of applying the model as described in section 4.

Metrics
- Accuracy and F1-score of stress classification are calculated.
- Metrics are reported as an average with a standard deviation of 10 runs.

Training Data
- The DTU dataset
  Physiological data from Empatica E4 from 28 subjects. The stress categorization comes from a timed puzzle task and the non-stress categorization from a resting phase.
- WESAD Schmidt et al. [2018]
  Physiological data from Empatica E4 from 15 subjects. The stress categorization comes from a Trier Social Stress Test (TSST) Kirschbaum et al. [1993] and the non-stress categorization from a resting phase and watching funny videos.
- AffectiveROAD dataset Haouij et al. [2018]
  Physiological data from Empatica E4 from 10 subjects (14 drives). The categorization comes from a subjective continuous stress evaluation of the driver during various driving scenarios.
- ADARP dataset Sah et al. [2022]
  Physiological data from Empatica E4 from 11 subjects diagnosed with alcohol use disorder (AUD). Data recorded in the wild. The stress categorization is from the subject tagging when they felt stressed.

Evaluation Data
- Running evaluations of the model have been done by leaving one training data set out as test data.
- The final evaluation will be done on a test set of the WristAngel data.
- The WristAngel dataset Olesen et al. [2023]
  Physiological data from Empatica E4 from 8 children with OCD. Data recorded in the wild. The stress categorization is from the child tagging when they felt stressed due to OCD symptoms.

Ethical Considerations
- Physiological data as used for the model, is very personal and should be handled in accordance with GDPR.

Caveats and Recommendations
- The final evaluation of the model has not been done yet, hence usefulness and considerations are subject to change.

Quantitative Analyses
To be written after testing on the WristAngel data.

7.2 Initial daily plan for WristAngel data analysis (Week 50, 2023: Monday to Friday)

Day 1 (Dec 11.): Understand the setup and how the data is formatted

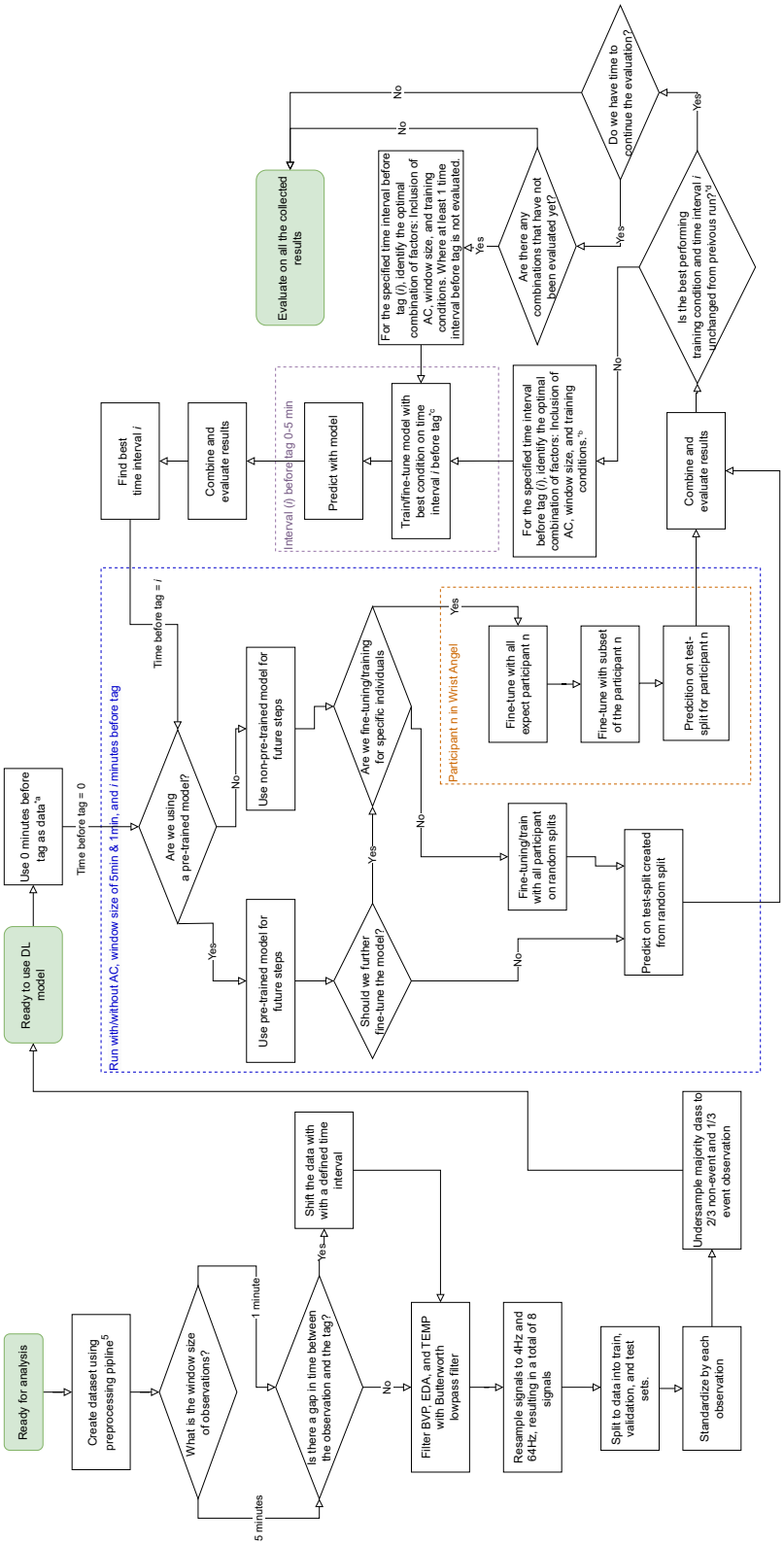
Day 2 (Dec 12.): Data preprocessing (this includes generating the datasets according to our models)

Day 3 (Dec 13.): The initial in-clinic data for movement

Day 4 (Dec 14.) – Day 5 (Dec 15.): Training and evaluating using WristAngel data

Additionally, a 3-day buffer where we could be there sporadically.

7.3 Flow chart



7.4 Example of how to evaluate model performance

Condition 1	Prediction time 0	1	⋯	5
Model 1				
Model 2				
⋮				

Condition 2	Prediction time 0	1	⋯	5
Model 1				
Model 2				
⋮				

(a) Initial step: Each model in each condition with prediction lead time 0.

Condition 1	Prediction time 0	1	⋯	5
Model 1	0.2			
Model 2	0.7			
⋮	⋮			

Condition 2	Prediction time 0	1	⋯	5
Model 1	0.3			
Model 2	0.3			
⋮	⋮			

(b) Second step: Pick the row/model and condition with the best performance. Run this model for each prediction lead time.

Condition 1	Prediction time 0	1	⋯	5
Model 1	0.2			
Model 2	0.7	0.4	⋯	0.75
⋮	⋮			

Condition 2	Prediction time 0	1	⋯	5
Model 1	0.3			
Model 2	0.3			
⋮	⋮			

(c) Thirds step: Pick column/prediction lead time with best performance. Run each model and condition with this prediction lead time.

Condition 1	Prediction time 0	1	⋯	5
Model 1	0.2			0.4
Model 2	0.7	0.4	⋯	0.75
⋮	⋮			⋮

Condition 2	Prediction time 0	1	⋯	5
Model 1	0.3			0.3
Model 2	0.3			0.6
⋮	⋮			⋮

(d) Final step: Performance has not improved.

Table 2: The steps of how we will evaluate the models are shown. The yellow cells are steps that will be taken. The green cells show the combination that created the best results and is the one we will continue. After the final step if time allows, we will continue with the best non-filled row or column.

The area of mental health and disabilities, such as obsessive-compulsive disorder (OCD), has received increased attention due to a large population, particularly adolescents, who suffer from the disorder. Through the application of machine learning, this project aims to support children with OCD in their daily lives, by providing timely warnings against OCD episodes. In the medical field, data scarcity is a common obstacle, and few studies have collected physiological data in the wild, especially concerning OCD in adolescents. However, a study called Wrist Angel, conducted at Børne- og Ungdomspsykiatrisk (Child and Adolescent Psychiatry) Center, Gentofte Hospital collected 8 weeks of daily physiological data from adolescents diagnosed with OCD. It showed that it is feasible to classify OCD events using real-world data. This thesis extends the work from the feasibility study by using transfer learning to improve the performance of predicting OCD events. It provides important insights into the efficacy of transfer learning with potential application to other domains.

Using a convolutional neural network (CNN) pre-trained on open-source stress datasets, stress events were predicted based on the physiological data, which are blood volume pulse (BVP), electrodermal activity (EDA), heart rate (HR), and skin temperature (TEMP).

In an exploratory phase, we tested various models, selected the best model architecture for the stress prediction task, and experimented with four datasets. By pre-training models with the best model architecture, we obtained a positive performance, with an F1-score up to 0.95, for predicting acute stress events for open-source datasets. Additionally, we improved performance using simulated data in some cases. However, when applying the pre-trained model to the in-the-wild OCD data from the Wrist Angel study, the model performance was inadequate. This was possibly due to a lot of noise, which is often an issue for in-the-wild time series data. We have nonetheless conducted tests indicating that fine-tuning on random splits with all participants works better than personal fine-tuning and that longer observation windows are not necessarily beneficial for predicting OCD events.

Our study contributes to the foundation for a new way of utilizing transfer learning to predict OCD events in adolescents based on physiological data.

Technical
University of
Denmark

Richard Petersens Plads, Building 324
2800 Kgs. Lyngby
Tlf. 4525 1700

www.compute.dtu.dk