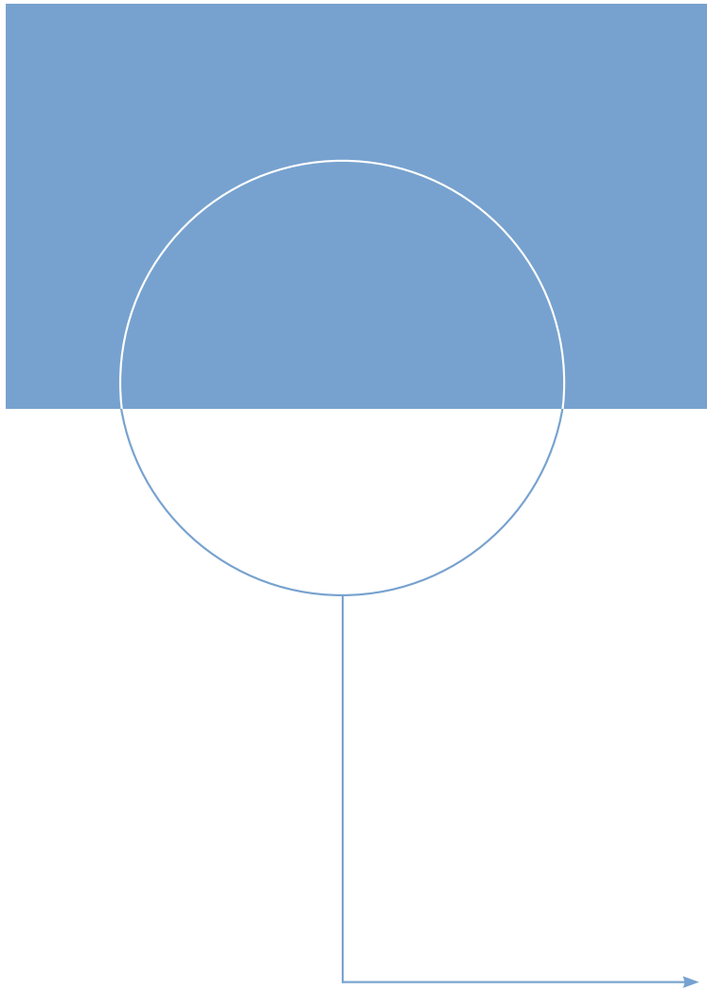


Master's thesis

2019

Aileen Hay

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Aileen Hay

Machine Learning Methods for Sleep-Wake Classification Using Two Body-Worn Accelerometers

June 2019

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

NTNU

Norwegian University of
Science and Technology

Machine Learning Methods for Sleep-Wake Classification Using Two Body-Worn Accelerometers

Aileen Hay

Computer Science
Submission date: June 2019
Supervisor: Kerstin Bach

Norwegian University of Science and Technology
Department of Computer Science

Abstract

Sleep is an important factor in protecting a person's physical and mental well-being. Consequently, many studies are conducted which focuses on preventing, diagnosing, and treating sleep disorders. A crucial part of these studies is the evaluation of subjects' sleep quality. This is often done through the use of polysomnography (PSG), the current gold standard for sleep quality evaluation. This is a quite expensive method that requires patients to spend at least one night in a sleep lab being monitored by various equipment. An experience often viewed as uncomfortable by patients. An alternative to PSG is to rely on the use of body-worn sensors to detect and record movement instead. The movement data can be analyzed to provide insights into sleep quality.

HUNT4 is the fourth iteration of the biggest population-based health study in Norway. It started in September of 2017 and was completed in February of 2019. The study relies on two body-worn Axivity AX3 accelerometer sensors, one placed at the lower mid back and one placed on the upper thigh, to gather accelerometer data from their participants. The collected data provides an indication of each participant's current activity level and health status. Sleep quality and sleep pattern information if also of huge interest to the HUNT4 study. However, to be able to obtain this information there is a need for a machine learning classifier that successfully detects sleep patterns from HUNT4 data. The main objective of our research is to create such a classifier.

In this thesis, we have conducted a structured literature review of related work in the field of sleep pattern detection, while also identifying potentials for improvements of current sleep pattern detection methods. We experimented using binary and multiclass classification, multi-view learning, and semi-supervised and unsupervised learning. During our experiments we implemented several ensemble methods for sleep pattern detection. Each method was trained and tested using data collected from 19 subjects diagnosed with sleep disorders. The performance of our proposed methods were compared to the performance of Decision Tree, Random Forest, and Extreme Gradient Boosting classifiers. All proposed methods outperformed these comparative counterparts. The best performance scores obtained was for the Supervised Multi-view with Agglomerative Hierarchical clustering method. The method reached accuracy, sensitivity, specificity, and g-mean scores of 94.51%, 99.13%, 85.16%, and 91.88%, respectively.

Sammendrag

Søvn er en viktig faktor for beskytte en persons fysiske og mentale trivsel. Derfor utføres mange studier som fokuserer på å forebygge, diagnostisere og behandle søvnforstyrrelser. En avgjørende del av disse studiene er evaluering av søvnkvalitet. Dette gjøres ofte ved bruk av polysomnografi (PSG), den nåværende gullstandarden for evaluering av søvnkvalitet. Dette er en ganske dyr metode som krever at pasienter tilbringer minst en natt i et søvnlaboratorium mens dem blir overvåket. En opplevelse som ofte blir betraktet som ubehagelig av pasientene. Et annet alternativ til PSG er å bruke sensorer festet på kroppen for oppdage og registrere bevegelse. Bevegelsesdataene kan analyseres for gi innsikt i søvnkvalitet.

HUNT4 er den fjerde iterasjonen av den største befolkningsbaserte helseundersøkelsen i Norge. Den startet i september 2017 og var ferdig i februar 2019. Studien bruker to kroppsplasserte Axivity AX3 akselerometer sensorer, en plassert på nedre del av ryggen og en plassert p låret, for å samle inn akselerometerdata fra deltakerne. De innsamlede dataene gir en indikasjon på hver deltakeres nåværende aktivitetsnivå og helsestatus. Søvnkvalitet og søvnmønsterinformasjon er også av stor interesse for HUNT4-studien. For å kunne skaffe seg denne informasjonen er det imidlertid et behov for en maskinlæring klassifikator som kan oppdage søvnmønstre fra HUNT4-data. Hovedmålet for vår forskning er å skape en slik klassifikator.

I denne oppgaven har vi gjennomført en strukturert litteratur gjennomgang av tidligere arbeid innen søvnmønster deteksjon samtidig som vi identifiserte forbedringsmuligheter for eksiterende søvnmønsterdeteksjonsmetoder. Vi eksperimenterte med binær- og multiklasseklassifisering, multi-view læring og halvovervåket og uåvervåket læring. I løpet av våre eksperimenter implementerte vi flere ensemblemetoder for søvnmønsterdeteksjon. Hver metode ble opplært og testet ved bruk av data samlet fra 19 personer diagnostisert med søvnforstyrrelser. Ytelsen til våre foreslåtte metoder ble sammenlignet med ytelsen til Decision Tree, Random Forest og Extreme Gradient Boosting klassifiseringsmetoder. Alle foreslåtte metoder overgikk disse tre metodene. De beste resultatene som ble oppnådd var for Åvervåket Multi-view med Agglomerative hierarkisk gruppering metoden. Metoden nådde nøyaktig-het, følsomhet, spesifisitet og g-mean verdier på henholdsvis 94.51%, 99.13%, 85.16% og 91.88%.

Preface

The project presented in this report was written for the Department of Computer Science (IDI) at the Norwegian University of Science and Technology (NTNU) in the spring of 2019. The scope and subject of the project was decided on in cooperation with our supervisor Kerstin Bach.

In addition, several people gave us a helping hand during our completion of this project. Therefore, we would first like to express our thanks and gratitude to our supervisor Kerstin Bach for her guidance and feedback throughout this project. A special thanks also goes out to Sverre Herland for his advice and help during our work this spring. Lastly, we would also like to thank Paul Jarle Mork, Atle Kongsvold, and Eivind Skarpsmo for collecting and providing the sleep data and annotations used in our work.

Table of Contents

List of Tables

List of Figures

Abbreviations

Acc	=	Accuracy
AHC	=	Agglomerate Hierarchical clustering
AL	=	Articicially labelled
ANN	=	Artificial neural network
CNN	=	Convolutional neural network
CoMV	=	Co-training with Multi-view
CoSV	=	Co-training with Single-view
DMLC	=	Distributed (Deep) Machine Learning Community
DT	=	Decision tree
EEG	=	Electroencephalographic
FN	=	False negative
FP	=	False positive
GNB	=	Gaussian naive Bayes
HMM	=	Hidden Markov Model
HUNT	=	The Nord-Trøndelag Health Study
IDI	=	Department of Computer Science
NTNU	=	Norwegian University of Science and Technology
OSA	=	Obstructive sleep apnea
PL	=	Professionally labelled
PLM	=	Periodic limb movement
PSG	=	Polysomnography
RF	=	Random forest
Sen	=	Sensitivity
SIDS	=	Sudden infant death syndrome
Spe	=	Specificity
SPT	=	Sleep period time
SuMV	=	Supervised Multi-view
SVM	=	Support Vector Machines
TN	=	True negative
TP	=	True positive
TST	=	Total sleep time
XGB	=	Extreme gradient boosting
XGBoost	=	Extreme gradient boosting

Chapter 1

Introduction

In this chapter we first introduce the background and motivation for our work. The specific goals and research questions are then described and lastly the outline of the remaining parts of the report will be presented.

1.1 Motivation

Sleep is an important part of everyday life. It plays a crucial role in preserving a person's health and well-being throughout the entirety of their lives. It is also essential for the protection of a person's physical and mental health. A lack of sleep will also impact people's safety and quality of life.

During sleep, the body is working to support healthy brain function while also working on healing and repairing heart and blood vessels. Studies have also shown that sleeps helps improve learning and recall in addition to helping with paying attention and improving decision making and problem-solving skills. On the other hand, sleep deficiency does the opposite and can diminishes these skills. It can also result in a person having a harder time controlling their emotions. As it becomes harder for a sleep deprived person to make sound judgment calls, the choices made by them can affect their own safety and also the safety of others.

A higher risk of diseases such as diabetes, stroke, heart and kidney disease have also been linked to sleep deficiency[1]. Furthermore, depression and suicide can become a risk. These are some of the reasons why many scientist are conducting studies with a focus on preventing, diagnosing, and treating sleep disorders. A vital part of these studies is to actually be able to evaluate subjects' sleep quality.

[1] https://www.nhlbi.nih.gov/health-topics/sleep-deprivation-and-deficiency, accessed: 2019-03-05

The current gold standard for sleep quality evaluation is polysomnography (PSG). This is a method requiring patients/subjects to spend at least one night sleeping in a sleep lab. During this time they have to be monitored by various equipment, some directly attached to them. The unfamiliarity of the surroundings and the extensive equipment often lead to subjects finding the environment uncomfortable to sleep in. Furthermore, PSG is quite a time-intensive, expensive method and some subjects, such as demented elderly and infants, are not able to tolerate it (Ancoli-Israel et al. (2003)). These are some of the reasons why other methods, such as actigraphy, is being pursued as an alternative to PSG.

Actigraphs can be defined as devices placed on the body to detect and record the movement of the subject (Ancoli-Israel et al. (2003)). They normally contain an accelerometer to detect the movement and a memory component to record it. The recording is often done continuously for several days and up to weeks at a time. This ability for continuous, long-term recording is one of actigraphy's main advantages, especially if compared to PSG. Furthermore, after the initial purchase of the equipment actigraphy has a low connected cost and does not require continuous monitoring by any technicians (Bourne et al. (2007)).

Another positive aspect connected to actigraphy is the fact that it can provide subjects with a more normal sleep setting. Subjects are not forced to sleep in a new and unfamiliar environment and can instead wear the sensors while sleeping in their own beds or during everyday activities. This will potentially result in a more accurate representation of a subjects true sleep pattern. Obtaining more accurate information will also in turn help give more accurate diagnoses. Actigraphy has already proven useful in diagnosing sleep disorders such circadian rhythm disorders (Ancoli-Israel et al. (2003)).

The Nord-Trøndelag Health Study (HUNT study)[2] is the biggest population-based health study in Norway. HUNT4[3] is the fourth iteration of the study and it started in September of 2017 and was completed in February of 2019 with the final numbers and quality approved data available in October of 2019. In addition to collecting biological samples (e.g. blood etc.) and self reported health data HUNT4 also includes data collection of physical activity and sleep through the use of two body-worn sensors. These sensors are placed on the upper thigh and lower back of each participant and worn for a week at a time. The accelerometer data obtained gives an indication of the participants' current health status. Sleep quality information is also a huge interest to the HUNT4 study.

Creating a classifier that successfully detect sleep patterns from data collected during the HUNT4 study will create a huge opportunity. At the moment the study is mainly relying on written information from the participants to get an insight into their sleep quality. Each participant is asked to fill out a questionnaire when they take part in the study and a few of the questions are about sleep. However, as the participants also were the sensors around the clock the collected data should also be able to provide information about their sleep. A sleep/wake classifier based on this data could help the study analyze the data of

[2]https://www.ntnu.no/hunt/om, last accessed: 2019-02-28

[3]https://www.ntnu.no/hunt4, last accessed: 2019-02-28

each subject and potentially help the study gain insight into their sleep quality.

The main objective of this research is to create a machine learning classifier for sleep pattern detection on sensor data that can be used in the HUNT4 study. The data collection set up and equipment used during this research is exactly the same as what is used in the HUNT4 study. This is done so that our classifier can be applicable to the study and so that the study is able to utilize our potential classier in the future. The proposed classifier should also be able to distinguish between wake and sleep instances with a high percentage of accuracy.

1.2 Goals and Research Questions

To be able to successfully create a method for sleep pattern detection we first need to obtain an overview of the work previously done in the field and get an understanding of the methods used.

Goal 1: To understand the state-of-the-art in the field of sleep pattern detection using machine learning methods.

- **RQ1**: Which machine learning techniques and data analysis methods have been used for sleep pattern detection on sensor data from body worn sensors?

Based on the knowledge gained from *RQ1* we will identify potential areas of improvement and implement and evaluate some selected methods.

Goal 2: To improve methods for sleep pattern detection on data collected from two-body worn accelerometer sensors (one at the lower back and one at the upper thigh).

- **RQ2**: How does multiclass classification affect the overall performance results for sleep pattern detection?

- **RQ3**: How do ensemble methods affect the overall performance results for sleep pattern detection?

1.3 Thesis Structure

The remaining chapters of this report can be described as follows.

- **Chapter 2 Background Theory** introduces background info related to actigraphy in general, current methods used for sleep detection and our previous work in the field.

- **Chapter 3 Related Work** presents the procedure used for the structured literature review conducted as a part of this research and provides summaries for papers currently a part of the state-of-the art research in the field of sleep detection.
- **Chapter 4 Beyond State-of-the-Art** presents the methods used during this research to potentially improve the sleep pattern detection system.
- **Chapter 5 Methodology:** gives an overview on the data collection process, the equipment used and the final data sets used in our research. It also presents the main procedure overview and provides more details on how the data was analyzed and structured.
- **Chapter 6 Experiments** introduces and explains the experiments conducted during this research.
- **Chapter 7 Results and Discussion** presents and discusses the results of the experiments conducted during this research.
- **Chapter 8 Conclusion and Future Work** gives an evaluation and conclusion of the work and findings presented in this report along with an overview of hypothetical areas of further work connected to the work described in the previous chapters.

Chapter 2

Background Theory

In this chapter we provide background information on the HUNT4 study, the differences between polysomnography and actigraphy and sensor placement used for sleep detection. We also introduce our previous work in the field and present relevant/basic machine learning algorithms suitable for sleep detection.

2.1 HUNT4 Study

The Nord-Trøndelag Health Study (HUNT study)[1] is the biggest population-based study in Norway. So far, the data has been collected through the completion of four sub-studies: HUNT1(1984-86), HUNT2 (1995-97), HUNT3 (2006-08) and HUNT4 (2017-2019). The collection consist of data pertaining to personal and family medical history and it's of use in several disease and health focused research projects.

HUNT4[2] started in September of 2017 and was completed in February of 2019 with the final numbers and quality approved data available in October of 2019. All inhabitants of Nord-Trøndelag, Norway, over the age of 19 received an invitation to participate in the study. Youths between the ages of 13 and 19 years old was invited to take part in the Ung-HUNT (Young-HUNT) study. As a part of the HUNT4 study, all participants was fitted with two Axivity AX3 accelerometer sensors, one at the upper left thigh and one at the lower back. The sensors were worn for a week by each of the participants before they were returned. The gathered accelerometer data was then analysed to measure the participants' overall activity level.

[1] https://www.ntnu.no/hunt/om, last accessed: 2019-02-28
[2] https://www.ntnu.no/hunt4, last accessed: 2019-02-28

2.2 Polysomnography

Polysomnography (PSG) is currently considered the gold standard for sleep quality evaluation. It is a non-invasive procedure and is normally performed in dedicated sleep clinics or hospitals. The evaluation procedure is conducted while the participant is sleeping (or attempting sleep). Through the use of different equipment, which is often attached to the participant, and observation trained sleep technicians are provided with a plethora of information about, for instance (Hirshkowitz (2015), Bourne et al. (2007)):

- Brain Waves (EEG)
- Body positioning
- Eye and body movements
- Blood Oxygen Levels
- Breathing Rates and Patterns
- Sleep stages
- Heart Rates and Rhythms

The data gathered through PSG can be used to diagnose several sleep disorders. It can also be used to examine how a patient's current treatment plans are working. When the procedure has been completed the sleep technicians examine the gathered information and evaluate and chart the results.

Despite the positive aspects connected to PSG it also have several disadvantages. According to Bourne et al. (2007) PSG is an expensive procedure. The setup and maintenance itself is quite costly. In addition, extensive resources are used to monitor the subject and equipment during the procedure and sleep technicians are needed to be present at all times. The procedure environment can also have a negative impact on the subjects. It is usually a novel setting for most of them and the procedure can therefore be viewed as a uncomfortable experience. These feelings usually lessen over time as the subject gets more familiar with the procedure and environment. However, since very few critical care studies are conducted over more than one day/night the subjects rarely get the opportunity to become comfortable.

Because of the cost and resources associated with PSG it is difficult to make expert-level sleep analysis widely available. However, a field that is emerging is the automatic annotation of sleep staging using machine learning. One example is the work of Biswal et al. (2017) who presents a deployed annotation tool for sleep staging called SLEEPNET. This tool uses a deep recurrent neural network, trained on PSG data collected from over 10 000 subjects, to make sleep annotations on PSG data. The tool achieves an annotation performance with an average accuracy of 85.76%. We want to note that even though we

found this concept of automatic annotation quite interesting it is not further addressed in this thesis, as our focus lies in the detection of sleep and wake phases for the setup used for objective measurement of physical activity in the HUNT4 study.

2.3 Actigraphy

Actigraphy is a non-invasive method that can be used for monitoring human movement/activity. An actigraph is a body-worn instrument/device that normally contains an accelerometer and a memory component. The accelerometer is what is used to detect movement and detected data is recorded in memory. The memory component is able to record data for several weeks at a time, 24-hours a day. After the initial purchase payment actigraphy has a low cost and can be used without constant continuous monitoring (Bourne et al. (2007)). The device can also be worn in a person's normal everyday settings.

Sensor placement

With the use of actigraphy, the most common sensor placement is the wrist. Cole et al. (1992) and Sadeh et al. (1994) introduced two of the earliest automatic scoring methods for sleep/wake detection. Both algorithms used wrist actigraphy and the proven successfulness of the algorithms has ensured that they are both still in use today. Several other wrist-actigraphy based methods can also be seen in more recent studies, such as Yeo et al. (2017) and Borazio et al. (2014). Furthermore, as many affordable wrist-worn activity monitors have become more and more available to the public in recent years the use of wrist actigraphy can be commonly seen in everyday settings.

Studies on using non-wrist actigraphy for sleep detection have also been conducted. Slates et al. (2015) and Zinkhan et al. (2014) both completed studies comparing hip and wrist sensor placement for actigraphic sleep detection. The results of both studies showed that wrist placement had superior performance results. However, it must be pointed out that the algorithms used to for evaluating hip actigraphy in both studies was originally developed for wrist placement. Another example of non-wrist actigraphy being used for sleep detection is Enomoto et al. (2009). Enomoto et al. (2009) introduces a sleep/wake pattern detection algorithm that uses activity intensity data gathered through the use of a waist-worn actigraph. The findings in the paper indicate that the results of their proposed algorithm is comparable to the results of more conventional actigraphy.

Using multiple actigraphs simultaneously has also been used in various studies. Lamprecht et al. (2015) executed a study where movement was recorded simultaneously from five differently placed sensors. The sensors were placed at the upper thorax, the left ankle, the left great toe, the left wrist and the left index fingertip. The results of the study indicated that if compared to single wrist actigraphy multisite accelerometry offers improved sleep/wake classification performance.

For this study we will continue our work with data obtained through the use/placement of two Axivity AX3 accelerometers. The sensors will have been placed at the upper thigh and the lower back. Duncan et al. (2018) conducted a study evaluating the validity of capturing 24-hour behavior profiles using the same sensors and placements. The conclusion of the paper being that the dual-accelerometer protocol demonstrated considerable promise for capturing movement patterns of free-living children and adults.

2.4 Previous work

In Hay (2018) we aimed to determine the viability of using machine learning algorithms for sleep pattern detection on sensor data collected from two body-worn accelerometer sensors (one at the lower back and one at the upper thigh). The same sensor equipment and sensor placement used to gather the data is also used for this research. By using the information gathered through a structured literature review we implemented five machine learning algorithms for this purpose: decision tree (DT), random forest (RF), articifial neural network (ANN), Gaussian naive Bayes (GNB), and extreme gradient boosting (XGB). Using the results of the literature review as inspiration we generated and combined features from personally manually labelled sensor data to be used for training and testing of the chosen supervised machine learning classifiers. The result from the research shows that DT, RF and XGB had the best performances with accuracies of 98,9%, 99,4%, and 98,2%, respectively. ANN and GNB had a worse performance with accuracies of 95,8% and 91,3%, respectively. The results of the research clearly indicate that the data gathered from the two sensors can be successfully used by the majority of the selected machine learning algorithm for the purpose of detecting sleep patterns.

2.5 Supervised Learning

Today, a wide array of algorithms have been used for sleep detection on sensor data. As mentioned previously, in the 90's Cole et al. (1992) and Sadeh et al. (1994) introduced proposed sleep detection algorithms which are still in use today. These algorithms, along with several of other algorithms used today, such as the algorithms used in the studies by Kim et al. (2013), Tudor-Locke et al. (2013) and Enomoto et al. (2009), can be viewed as more traditional activity count-based methods. Nonetheless, the selection of methods in use today consists of a wide variety and is not limited to the more traditionally viewed methods.

During recent years focus on using supervised machine learning methods for automatic sleep/wake detection has increased. Supervised learning is a type of machine learning method where large amounts of labelled data is used consisting of observations/training data and labels/target values. This data is used to train a model that will be capable of clas-

sifying new instances of data. In terms of supervised learning for sleep/wake classification through the use of actigraphy, a model will be trained based on labelled accelerometer data. Some examples of supervised learning algorithms that have shown promise in automatic sleep/wake classification are: decision tree, random forest, and extreme gradient boosting (Tilmanne et al. (2008), Yeo et al. (2017), Khademi et al. (2018)).

2.5.1 Decision Trees

The decision tree (DT) algorithm is a classification method that creates a map (a decision tree) between the observed attributes of an data instance (input) to the predictions about its class (label) (Tan et al. (2014)). There are three types of nodes contained in a decision tree:

- *Root node:* a node with no incoming edges but with zero or more outgoing ones.
- *Internal nodes:* a node with one incoming edge and two or more outgoing ones.
- *Leaf or terminal nodes:* a node with one incoming edge and no outgoing ones.

Figure 2.1 shows an example of a decision tree used for deciding if you should mow the lawn (cut the grass) or not. As seen in the figure each leaf node in a decision tree has a class assignment. In the ”Mow Lawn” example the classes are either ”Yes” or ”No”. The root and internal nodes all use attribute test conditions to separate between different features instances. For instance, the internal node in Figure 2.1 separates between rainy and non-rainy days with the attribute *Weather*.

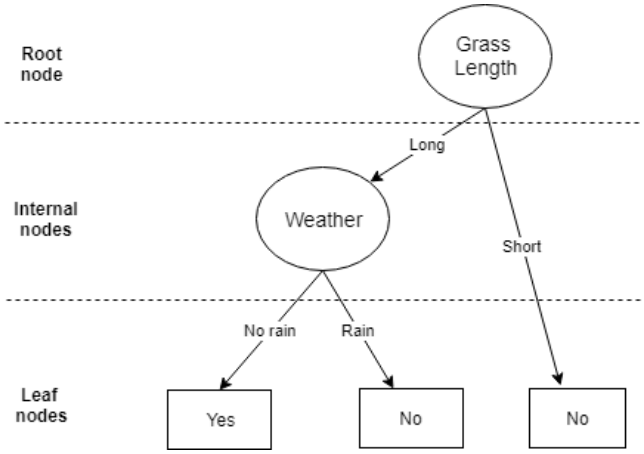


Figure 2.1: Decision tree for the concept ”Mow Lawn”.

During each recursive step of the decision tree building procedure an attribute split must be done. There are several different measures used to select the best split, such

as Gini, Information Gain, and classification error. The tree building process stops only when the stopping criteria is met. Algorithm 1 shows a pseudocode for the decision tree algorithm.

Algorithm 1: Decision tree algorithm.

```
Let E be the training set and F the attribute set.
DecisionTree(E, F)
    if Stopping conditions of E and F are true then
        Create leaf node l.
        Label l with the majority class of E.
        Return l.
    else
        Create new node r.
        Find best attribute split S to split E.
        Let V equal all possible values of S.
        for v in V do
            Set E_v as the subset of E that has value v.
            child = DecisionTree(E_v, F).
            Add child as a descendent of r and label the edge (r → child) as v.
        end
    end
    Return r.
```

2.5.2 Random Forest

Random forest (RF) is a supervised machine learning algorithm that relies on the use of several decision trees to make predictions (Yeo et al. (2017)). The first step of the algorithm is to create n random subsets of the original data set. From there on, a decision tree is created for each of these subsets resulting in n decision trees. After the creation of the decision trees an instance is labelled by having all decision trees make a prediction and then using a majority voting scheme to decide on the final prediction. A pseudocode of the algorithm is shown in Algorithm 2.

2.5.3 Extreme Gradient Boosting

Extreme gradient boosting (XGBoost) is a supervised learning method that is a variant of the more commonly known boosting method. Boosting is a machine learning ensemble method that adaptively makes changes to the distribution of training examples for the purpose of giving base classifiers a greater focus on harder to classify examples (Tan et al. (2014)). Several different implementations of the boosting algorithm have obtained a significant amount of popularity in machine learning. One of these implementations is the

Algorithm 2: Random forest algorithm.

Let n be the number of decision trees.
for *i = 1 to n* **do**
  Sample the original data to create a training set T_i of size k.
  Use the training set T_i to create a decision tree f_i.
end
$f^*(x) = \arg\max_y \sum_i \delta(f_i(x) = y)$.

$$(\delta(\cdot) = \begin{cases} 1 \text{ if its argument is true} \\ 0 \text{ otherwise} \end{cases})$$

XGBoost method.

XGBoost[3] is designed for speed and model performance and it is an implementation of the gradient boosted decision tree algorithm. Gradient boosting is a boosting algorithm where the main idea is to combine more and more simple models together so that the overall model becomes stronger[4]. A final prediction is made after all the models have been added sequentially and no more improvement can be done. A gradient descent algorithm is used to minimize the loss from the addition of each new model, thus the name gradient boosting.

The XGBoost algorithm was created by Tianqi Chen[5] as a part of a research project of the Distributed (Deep) Machine Learning Community (DMLC) group. It can be used for regression as well as classification.

2.5.4 k-Nearest Neighbour

An additional supervised learning method that can be used for sleep detection is the k-Nearest Neighbour algorithm. This method bases its predictions on the similarity between instances and can be used for both classification and regression. In a classification task a new instance is classified by finding the n nearest, i.e. most similar, neighbours. The neighbours consists of the already classified instances. The most common class among the nearest neighbours is then set as the label of the new instance.

[3]https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/, accessed: 2019-02-03

[4]https://explained.ai/gradient-boosting/faq.html, accessed: 2019-02-28

[5]https://homes.cs.washington.edu/∼tqchen/2016/03/10/story-and-lessons-behind-the-evolution-of-xgboost.html, accessed: 2019-02-03

2.6 Unsupervised Learning

One of the main disadvantages of supervised learning is that it requires labelled data. This means that if no labels are available a lot of time and resources need to be invested to obtain the large amount of labelled data necessary for training and testing the models. With regards to supervised learning for sleep detection this means PSG is often necessary to obtain the labelled data and, as we have already discussed, PSG comes with several disadvantages. Another option is to use unsupervised learning instead.

Unsupervised learning is a type of machine learning that solely relies on data that is unclassified and unlabelled. The goal of unsupervised learning is to create a model of the data's underlying structure in order to learn more about it[6]. EL-Manzalawy et al. (2017) use classification via clustering and their work is one example of unsupervised learning being used for sleep detection.

2.6.1 Clustering Methods

Clustering methods are some of the most commonly used unsupervised learning methods. The goal of clustering is to find meaningful data groups/clusters. The definition for what constitutes a meaningful cluster can vary depending on the task at hand.

The most common distinction between types of clustering is partitional versus hierarchical. With partitional clustering the data set is divided into non-overlapping subsets/clusters and each data instance can only be found in one cluster. However, with hierarchical clustering the data is divided into a set of nested clusters. The nested clusters are organized into a tree where clusters are allowed to have subclusters and the root of the tree contains all instances (Tan et al. (2014)).

K-means clustering is a partitional clustering method. The goal of the method is to try and find a user-specified number (k) of clusters. The clusters are created by first selecting k initial centroids which each represents a cluster. Each data instance is then added to the cluster represented by the closest centroid. After all instances are clustered the centroids are updated based on the instances connected to each cluster. The two last steps are then repeated until the centroids no longer change. There exist several variations of the k-means algorithm. The pseudocode for the basic k-Means algorithm can be seen in Algorithm 3.

Another group of clustering methods is the agglomerative hierarchical clustering approach. This approach represents a collection of clustering methods that creates a hierarchical clustering by assigning each instance to their own singeton cluster and then repeatedly merging the two closest clusters together until only one cluster remains (Tan et al. (2014)). The pseudocode for the basic agglomerative hierarchical clustering algorithm can be seen in Algorithm 4.

[6] https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/, accessed: 2019-02-08

Algorithm 3: Basic k-Means algorithm (Tan et al. (2014)).

Select k data instances as initial centroids.
while *Centroids keep changing* **do**
  Create k clusters by assigning each data instance to the closest centroid.
  Recalculate the centroid of each cluster.
end

Algorithm 4: Basic agglomerative hierarchical clustering algorithm.

Assign each data instance to their own separate cluster.
while *Number of clusters is larger than 1* **do**
  Calculate the proximity between clusters.
  Merge the two closest clusters into one cluster.
end

2.7 Evaluation of Methods

Evaluation of an algorithm is always an important step to take to achieve an accurate understanding of the performance of the algorithm. There exists a variety of methods for evaluating, one being the confusion matrix. With regards to evaluating the performance of a classification algorithm, the confusion matrix is quite suitable. It can be used to calculate sensitivity, specificity and accuracy (Tilmanne et al. (2008)). Table 2.1 illustrates a confusion matrix for sleep/wake classification. G-mean, which is based on sensitivity and specificity, is another metric of interest when evaluating binary classification performance (Tang et al. (2009)).

Actual Class	**Predicted Class**	
	Sleep	*Wake*
Sleep	TP	FN
Wake	FP	TN

Table 2.1: Confusion Matrix: TP = True positive, FN = False negative, FP = False positive, TN = True negative.

Accuracy (Acc): is the percentage of instances correctly classified by the algorithm.

$$Acc = \frac{\text{TP + TN}}{\text{TP + TN + FP + FN}} \tag{2.1}$$

Sensitivity (Sen): is the percentage of actually positive instances correctly classified as

positive.

$$Sen = \frac{\text{TP}}{\text{TP + FN}} \tag{2.2}$$

Specificity (Spe) is the percentage of actually negative instances correctly classified as negative.

$$Spe = \frac{\text{TN}}{\text{TN + FP}} \tag{2.3}$$

G-Mean is the geometric mean of specificity and sensitivity. The value is in the range of 0-1 with 1 being the optimal result.

$$G - Mean = \sqrt{\text{Sen} * \text{Spe}} \tag{2.4}$$

Chapter 3

Related Work

In this chapter we introduce the search procedure and initial results from our structured literature review along with an explanation of the quality assessment used. Following this, a brief summary of each of the papers selected as a part of the ”Related Work” group will be given. Lastly, a summary of any information extracted from the introduced papers that is found relevant to our research will be presented.

3.1 Structured literature review

To obtain an overview of the current state-of-the-art in the field of sleep pattern detection we performed a structured literature review. This section gives brief overview of the steps taken during this procedure.

3.1.1 Search Procedure

To make the searching as efficient and structured as possible we created search term groups. These search terms was used to find relevant literature through five main sources. These sources were:

- IEEE Xplore Digital Library[1]
- SpringerLink[2]
- Wiley Online Library[3]

[1] https://ieeexplore.ieee.org/Xplore/home.jsp
[2] https://link.springer.com/
[3] https://onlinelibrary.wiley.com/

- Engineering Village[4]
- Google Scholar[5]

The final selection of search terms used can be viewed in Appendix A. In addition, several additional papers found outside the above sources was also added to the literature review.

3.1.2 Search Results

After finishing the literature search, each paper was given a brief quality assessment and review, with regards to its relevance to our research goals. Any papers found lacking in quality was removed and the remaining was grouped based on the main focus of the paper in relationship to the work in this thesis. The two main groups where:

- Related Work: contains papers that have a focus on different algorithms and methods used or could potentially be used to detect sleep patterns.
- Background: consists of papers with a main focus on background information, such as sensor placement, diagnosing sleep and actigraphy in general.

The specific criteria used for our quality assessment can also be viewed in Appendix A.

3.2 Related Work

This section presents related research to the work presented in this thesis. They are divided into four subgroups: Sensor Placement, Supervised Learning, Unsupervised Learning, and Miscellaneous.

3.2.1 Sensor Placement

In this section we focus on papers that discuss the placement of the actigraph(s) used for automatic sleep detection.

Lamprecht et al. (2015) evaluates the validity of utilizing multisite tri-axial accelerometry for improving sleep/wake classification. Data used in the research was collected from 24 subjects with obstructive sleep apnea (OSA) aged between 6 and 15 years old. The severity of the diagnosed disorder ranged from healthy to severe. Each subject underwent PSG while simultaneously wearing a custom multisite accelerometry system to record movement. The system recorded motion from the upper thorax, the left wrist and index

[4]https://www.engineeringvillage.com/
[5]https://scholar.google.no/

fingertip, and the left ankle and great toe. To calculate the classification performance of both single-wrist and multisite accelerometry quadratic discriminant analysis was performed. Compared to single-wrist actigraphy, the results clearly shows that utilizing multisite accelerometry improved sleep/wake classification performance. These findings give a good indication that using multisite accelerometry might offer improved performance for sleep/wake classification in general.

Tudor-Locke et al. (2013) investigates the validity of using an already published sleep detection algorithm, along with two additional refinements (algorithms), for data collected through waist-worn actigraphy. The first algorithm is Sadeh's algorithm (Sadeh et al. (1994)) which can be described as a traditional activity count-based method and was originally developed for wrist actigraphy. The second algorithm makes adjustment to the sleep-period time estimation of Sadeh's algorithm by applying the sensor's inclinometer function. Furthermore, the third and last algorithm build further unto this adjustment by also modifying an already existing non-wear algorithm. All three algorithms were implemented and tested and the results showed that sleep time was significantly overestimated by the two first algorithms. On the other hand, Algorithm 3 had precise estimations that where within the expected values and had only a mean difference of 2 minutes.

Enomoto et al. (2009) presents a sleep/wake pattern detection algorithm based on activity intensity data collected by a waist-worn actigraph (the Lifecorder PLUS). 31 healthy subjects underwent PSG while simultaneously wearing the Lifecorder PLUS. While being worn the actigraph detects and saves an activity score for every 2 minute epoch. For the sleep/wake classification algorithm a discriminant score is calculated based on the activity intensity data for each 2 minute target epoch along with the intensities of the previous and following epochs. This score is then used to classify the target epoch as either sleep or wake. The performance results of the algorithm showed a mean agreement rate of 86.9%, mean sensitivity of 89.4% and a mean specificity of 58.2% when compared to corresponding PSG-based data. For the individual sleep stages the agreement rates were at 89% or above for Stage REM, Stage 2, and Stage 3+4. The agreement rate for Stage 1 was only at 60.6%. These findings shows that the LifeCorder PLUS waist-actigraphy along with the proposed algorithm is comparable to more conventional actigraphy.

3.2.2 Supervised Learning

In this section we focus on papers presenting work utilizing supervised learning algorithms for sleep detection.

Tilmanne et al. (2008) examines the performance of two new scoring algorithms used for distinguishing between sleep and wake states in infants through actigraphy. One of the algorithms is based on applying artificial neural networks (ANNs) and the other on decision trees (DTs). For validation, the performance of the algorithms is compared to the performance of two known sleep detection algorithms: Sazanov's algorithm and Sadeh's

algorithm. The results shows that the ANN and DT algorithms outperformed Sazanov and Sadeh's algorithms with a highest accuracy of 80.3% and 82.1%, respectively. The findings in the paper shows that the two new algorithms are robust and suggests that both ANNs and DTs can be suitable for further use in the context of clinical sleep research.

Yeo et al. (2017) investigated the possibility of using machine learning algorithms to create a model for automated sleep/wake classification on accelerometer data collected through wrist actigraphy. Sleep data was obtained from 36 subjects who spent one night in a sleep lab while accelerometer and PSG data was recorded simultaneously. The authors implemented and tested five supervised machine learning methods: random forest, bagging, KStar, random committee, and random subspace. Each method was used to classify the data to one of the following sleep stages: wake, REM, light, and deep. For validation the results was compared with the PSG scoring results. The results of the paper showed that the sensitivity scores was quite low (between 50 and 80 %), while specificity and accuracy was above 90 %. The conclusion of the paper being that the suggested algorithms could be efficiently applied for automatic sleep stage scoring.

Orellana et al. (2014) presents an artificial neural network (ANN) model for sleep/wake classification based on wrist actigraphy collected from adolescents during night time. The data used for training and testing was collected in 1 min epochs and the sleep and wake instances was balanced to improve system training. For data analysis an 11-minute sliding window was used. The performance results of the ANN classifier was compared against the performance results of Sadeh's algorithm. The results showed that the ANN classifier had higher performance scores than Sadeh's algorithm. The sensitivity score was at 97.6%, the specificity score 73.4%, and the accuracy score was at 92.8%.

Khademi et al. (2018) examines whether or not personalized machine learning classifier can compete against the performance of generalized machine learning classifiers. The authors implemented and tested five commonly used machine learning methods: extreme gradient boosting, AdaBoost, naive Bayes, regularized logistic regression, and random forest. For the personalized classifiers the models was trained on individual data and for the generalized classifiers they were trained on population data. The results in the paper showed that extreme gradient boosting had the strongest overall performance results. Furthermore, the results also showed that the personalized classifiers clearly performed at the same level as their generalized counterparts. This shows that creating reliable personalized sleep-wake classifiers for accelerometer data is a feasible option.

Phan et al. (2019) proposes a convolutional neural network (CNN) based joint classification-and-prediction framework for automatic classification of sleep stages. The sleep stages consists of five different stages: W, N1, N2, N3, and REM. The proposed framework takes in a single epoch and determines its label (classification) while at the same time determining its neighboring epochs' labels (prediction). To evaluate the performance of the framework two public data sets consisting of PSG recordings are used instead of data gathered through actigraphy. The results shows that the proposed framework had a clas-

sification accuracy of 83.6% and 82.3%, while it also outperforms existing deep-learning approaches.

Granovsky et al. (2018) proposed two novel methods for sleep/wake detection based on 1-dimensional Deep Convolutional Neural Networks (CNNs). The first method is a sequential CNN, while the second is based on Multi-Task Learning. In addition, the sleep and wake states are expanded to also include the additional states of ”Falling asleep” and ”Siesta” (resting state). Data for the study was collected through the use of wrist-worn actigraphy over a period of 12 weeks from 25 subjects with chronic Cluster Headache (CH). Both proposed CNN methods was implemented and tested and then compared to the performance of two forms of Recurrent Neural Networks (RNN): bi-directoinal Gated Recurrent Unit (GRU) and Long Short Term Memory (LSTM). In addition, the performance of the proposed methods were also compared to the performance of standard multilayer perceptron. The results of the comparison show that the proposed method clearly had higher accuracy scores and faster convergence rates than their comparative counterparts. Furthermore, the findings also give an indication that the two proposed CNN methods can accurately detect each of the four states.

3.2.3 Unsupervised Learning

In this section we focus on papers presenting work using unsupervised learning algorithms for sleep detection.

EL-Manzalawy et al. (2017) introduces a new proposal for developing reliable sleep-/wake classification models. The proposed approach is based on using unlabelled data gathered through wrist actigraphy combined with domain knowledge heuristics and unsupervised learning. To specify, the papers implemented and tested four different unsupervised clustering methods for classification: k-means, fuzzy c-means, Gaussian mixture using full covariance matrix, and Gaussian mixture using diagonal covariance matrix. The performance results of the clustering methods was compared to the performance of four supervised learning methods. The supervised learning models were trained based on labelled actigraphy and PSG data. The results in the papers shows that k-means clustering had the best performance among the clustering methods with results comparable with the performance results of the supervised learning methods. With regards to the supervised learning methods, Gaussian Naive Bayes (GNB) had the best overall estimates for the sleep parameters.

Li et al. (2018) proposes an unsupervised algorithm that uses a Hidden Markov Model (HMM) to automatically classify sleep/wake epochs. The algorithm is evaluated by doing an epoch-by-epoch comparison between the performance of the Actiwatch software and the performance of the proposed algorithm. For the evaluation an Actiwatch dataset comprised of data collected from 82 2-year-old toddlers was used. The results in the papers suggests that the proposed unsupervised algorithm outperforms the Actiwatch software

while it simultaneously overcomes the limitations of current ad hoc methods.

3.2.4 Non-traditional Methods

In this last section we present papers that introduces non-traditional algorithms for sleep detection.

Borazio et al. (2014) introduces a new sleep/wake detection algorithm for tri-axis accelerometer data. The proposed algorithms is based on a principle of Estimation of Stationary Sleep-segments (ESS). Accelerometer and PSG data was collected from 42 subjects aged between 28 and 86 years. All subjects suffered from a form of sleep disorder. The proposed method was compared to the performance of two traditional sleep detection methods: Cole et al.'s and Oakley's algorithm. However, unlike these traditional methods the ESS algorithm does not rely on activity count to classify epochs. Instead, the algorithm relies on the presence of long periods of idleness. These periods can be seen in the accelerometer data as flat horizontal signals. The results of the comparison of methods showed that the proposed new ESS algorithm performed slightly better than its comparative counterparts with an overall median accuracy of almost 79%.

van Hees et al. (2018) evaluated the possibility of estimating sleep parameters from wrist-worn raw accelerometer data without the presence of sleep diaries. For this purpose the authors introduces their own heuristic algorithm. The proposed algorithm is based on the variance of the z-axis angle and makes some assumptions on the nature of sleep interruptions. To evaluate the performance of the algorithm its ability to detect the sleep period time window (SPT-window) was compared to sleep diaries and PSG. In comparison with sleep diaries of men and women, the results showed that the algorithm's detected SPT-window was longer by 10.9 and 2.9 minutes, respectively. Furthermore, they also used the c-statistic, also known as the area under the ROC curve, to evaluate the results. It is a measure used in logistic regression that ranges from 0.5 to 1.0, with higher values indicating better fit of the model. When compared to PSG data of healthy and clinic-based sleepers, the mean C-statistic for detection of the SPT-window was 0.82 and 0.86. Overall, these findings demonstrates the usefulness of the proposed heuristic algorithm for studies where sleep diaries are absent. The code for the proposed algorithm has also been implemented and can be found in an open source R package called GGIR[6].

3.3 Summary

This section gives a summary of the most important relevant information obtained from the papers described in the previous section.

[6]https://cran.r-project.org/web/packages/GGIR/

3.3.1 Data Collection

The sleep data used for sleep detection was obtained through the use of actigraphy in all papers except Phan et al. (2019). More information on some of the specific sensor placement used has already been provided previously in Section 2.3. The accelerometer data was also often collected simultaneously with PSG.

Subjects

The subjects used for data collection in the papers consisted of subjects of various ages and health status. The age groups ranged from infants to older adults and healthy subjects was only primarily used in five of the papers. All remaining papers had data collected from subjects with sleep disorders, subjects from various different groups, or not specified. The number of subjects used in the papers also varied. The lowest number of subjects used being 20 and the highest number 354.

Table 3.1 gives on overview of specific subject information concerning the papers in Section 3.2.2 and 3.2.3: the papers focusing on machine learning methods.

Paper	**Subjects**	**Age group**	**Health status**
Tilmanne et al. (2008)	354	Infants	Varied: Healthy term, preterm, siblings of SIDS, and infants with apparent life-threatening events
Yeo et al. (2017)	36	NA	NA
Orellana et al. (2014)	119*	Adolescents	Healthy
Khademi et al. (2018)	54	Varied: Adults, older adults, not specified	Varied: Healthy, not specified
Phan et al. (2019)	20, 200	25-34 years old, 18-76 years old	Healthy, healthy*
Granovsky et al. (2018)	25	Unknown	Chronic cluster headache
EL-Manzalawy et al. (2017)	37	Varied: older adults, not specified	Varied: insomnia, baseline sleep disorder, sleep restriction in healthy subjects
Li et al. (2018)	82	2-year old toddlers	Healthy

Table 3.1: Subject information (*presumed).

Amount of Data

Another important aspect to take into account is the amount of data used for training and testing of the machine learning methods used in the papers. This information will provide an overview of the amount of data necessary to achieve a stable machine learning classifier. Table 3.2 shows a summary of this information.

Paper	Subjects	Nights /subject	Total minutes	Data collection	Labelled
Tilmanne et al. (2008)	354	1	168 500	Actigraphy, PSG	Yes
Yeo et al. (2017)	36	1	17 280*	Actigraphy, PSG	Yes
Orellana et al. (2014)	119*	2	64 102	Actigraphy, PSG	Yes
Khademi et al. (2018)	54	3*	81 000*	Actigraphy, PSG	Yes
Phan et al. (2019)	20, 200	2, 1*	18720*, 96 000*	PSG	Yes
Granovsky et al. (2018)	25	84	NA	Actigraphy	Yes
EL-Manzalawy et al. (2017)	37	3-11	114 745	Actigraphy, PSG	Yes
Li et al. (2018)	82	7	625 040*	Actigraphy, Sleep Diaries	No

Table 3.2: Amount of data used in papers focusing on machine learning (*presumed).

3.3.2 Machine Learning Methods

Several papers in Section 3.2 proposes the use of machine learning methods for sleep detection. The machine learning methods that performed best in the different papers are as follows:

- Decision Tree (DT)
- Artificial Neural Network (ANN)
- Random Forest (RF)
- Extreme Gradient Boosting (XGB)

- Gaussian Naive Bayes (GNB)
- K-means clustering
- Hidden Markov Model (HMM)
- Convolutional Neural Network (CNN)

In the majority of the work machine learning methods carry out binary sleep/wake classification. The exceptions are the work of Phan et al. (2019), Granovsky et al. (2018) and Yeo et al. (2017). Phan et al. (2019) and Granovsky et al. (2018) both use CNNs to distinguish between multiple different sleep states, while Yeo et al. (2017) does the same but with the use of RF (and additional methods). However, it should be noted that RF has also been used for binary sleep/wake classification in other work (Khademi et al. (2018)).

3.3.3 Data Processing

Many machine learning methods require pre-processing of data. Pre-processing methods often used in the related work includes filtering, segmentation and feature engineering. This section summarizes the use of these pre-processing methods as described in the papers presented before.

Filtering

Only five of the papers introduced in this chapter describes use of filtering in their data pre-processing. These papers are Lamprecht et al. (2015), Tudor-Locke et al. (2013), Yeo et al. (2017) Phan et al. (2019) and Orellana et al. (2014). The last three paper have a machine learning focus.

Lamprecht et al. (2015) uses two separate filter protocols. The first one is a fifth-order low pass Butterworth filter with a cutoff frequency of 2Hz. The second protocol is a 10th order bandpass Butterworth filter with the cutoff frequency set between 2 and 12 Hz. Yeo et al. (2017) also uses a fifth-order Butterworth filter, with the bandpass filter's cut-off frequency set between 0.25 Hz to 3 Hz.

Tudor-Locke et al. (2013) only specifies using a the low-frequency extension filter during their pre-processing. Orellana et al. (2014) also only states they used filtered to remove low-power noise from their data without providing specifics.

Lastly, Phan et al. (2019) used a frequency-domain filter bank for frequency smoothing of their PSG data. However, it should be noted that since they did not use accelerometer data it is not as relevant to our work.

Segmentation

For most of the papers described (9 out of 13) the data used for sleep detection was segmented into 30 second epochs. PSG data also most commonly comes in this same format. Choosing to segment their own data into an epoch size of 30 seconds was therefore perceived to be done by the authors to enable easier comparison with PSG data.

When evaluating an epoch for classification/feature generation, only 3(4) papers does not take into considering the surrounding epochs as well. These are Borazio et al. (2014), Lamprecht et al. (2015), and Yeo et al. (2017). van Hees et al. (2018) can also partially be included in this group since they don't use epochs at all in their calculations.

Feature Engineering

The following is an overview of the feature selection used in the papers presenting work using machine learning. Phan et al. (2019) in not included as its features are based on PSG data and not accelerometer data.

Tilmanne et al. (2008) uses 25 features calculated based on a sliding window of 10,5 minutes with the 30 second target epoch in the center. The features included, but was not limited to, max, min epoch activity and mean activity value.

Yeo et al. (2017) has a feature selection consisting of: mean, standard deviation, correlation, kurtosis, crest factor, skewness, zero crossing, entropy, band energy, and Spectral Flux. The features was calculated for each 30 second target epoch,

Orellana et al. (2014) uses 34 features calculated based on a sliding window of 11 minutes with the 1 minute target epoch in the center. This included, but was not limited to, median, minimum value, and raw and logarithm activity levels.

Khademi et al. (2018): uses 39 features calculated based on a sliding window of 10,5 minutes with the 30 second target epoch in the center. The features included: mean, sum of values, zero crossings, maximum value, standard deviation, kurtosis, skewness, coefficient of variation, inter quartile range, peak-to-peak amplitude, signal power, time above threshold, peak intesity, 21 normalized actigraphy measurements and 10th, 20th, 50th, 75th, and 90th percentiles.

Granovsky et al. (2018) has a feature selection consisting of a 721 dimensional feature vector extracted from a sliding window of 6-hours with the 30 second target epoch in the center.

EL-Manzalawy et al. (2017) has a feature selection/data representation based on a sliding window of 10,5 minutes with the 30 second target epoch in the center. Two different data representations are used: Binarized activity counts (BAC) and Normalized activity counts (NAC) where both counts are based on the 21 activity counts from the 21 epochs found in the sliding window.

Li et al. (2018) represents the data as activity counts of 1 minute epochs.

Chapter 4

Beyond State-of-the-Art

In this chapter, we describe arousal classification, semi-supervised learning and multi-view learning, and how these techniques can potentially improve performance results for sleep pattern detection.

4.1 Arousal Classification

An arousal during sleep can be defined as a fast change in EEG frequency (the electrical activity in the brain), which can be followed by changes such as a rise in heart rate and limb movements or changes in body posture (Halász et al. (2014)). If an arousal happens it does not mean a person is awake. It typically represent a person's change from a deep sleep state to a light sleep state and increased occurrences of arousals throughout a night can prevent a person from obtaining a solid/deep night's sleep[1].

Since the 1980's the role of arousals in sleep have gained more and more interest from scientists as increasing evidence indicate that arousals are deeply involved with the functional changes that accompany sleep disorders (Halász et al. (2014)). Using sleep data to accurately indicate the occurrence of arousals can therefore be said to be of huge interest to many scientists and a step beyond normal sleep/wake classification.

Fonseca et al. (2013) investigates the impact arousals have on the performance of actigraphy-based sleep/wake classification. The findings presented in the paper show that the occurrence of arousals had a significant effect on the number of misclassified epochs as observations showed that body movements would sometimes follow arousals during sleep. This could lead to sleep epochs following an epoch containing an arousal being instead classified as wake. The conclusion of the paper being that unless arousals and the

[1] https://www.verywellhealth.com/arousal-during-sleep-3014849, accessed: 2019-02-27

movements connected to them are automatically distinguished from wake any actigraphy-based sleep/wake classifier will be limited in their performance whenever arousals are present.

Following this conclusion, developing a machine learning classifier that distinguishes between sleep epochs, wake epochs and sleep epochs containing arousals could be a natural next step. This would entail obtaining labelled sleep data that also contains an overview of when any arousals occurred during the night. The classification problem would also change from a binary classification problem to a multiclass classification problem.

Binary classification is the classification of an instance as one of two classes. This is what we did in our previous work with sleep detection (Hay (2018)), using the classes wake and sleep. As seen in the previous chapter, it is also what is primarily done when using machine learning for sleep detection. Multiclass classification is the classification of an instance as one of three or more classes. With regards to the methods used by us previously (DT, RF, ANN, GNB and XGB) all of them can be altered into solving a multiclass problem instead of a binary one. However, additional features that better represent the occurrence of arousals might need to be added to the feature set for the classification to perform as needed.

4.2 Semi-supervised Learning

As mentioned previously, supervised learning relies on both input and output data (labelled data) to find a function that uses the input to approximate the output. On the other hand, unsupervised learning rely solely on input data (unlabelled data) to find an representation of the data structure. Semi-supervised learning is a type of machine learning that falls between supervised and unsupervised learning as it relies on both labelled and unlabelled data.

Typically, semi-supervised algorithms us a small amount of labelled data to train a classifier witch is then used to classify a large amount of unlabelled data (Aridas and Kotsiantis (2015)). For sleep detection, using semi-supervised learning can significantly minimize the amount of time and resources needed for obtaining a necessary amount of labelled sleep data.

Semi-supervised learning can be divided into the subgroups of single-view and multi-view algorithms. Single-view is where the semi-supervised algorithm uses a single feature set (view) to create a model/classifier. Multi-view is when the algorithm uses two different feature sets (views) to create two different models/classifiers.

4.2.1 Self-training

Self-training is a semi-supervised single-view learning method. The method starts by training a classifier on available labelled data. The classifier is then used to predict labels

for all instances of unlabelled data. Any instances with a prediction confidence above a given threshold is added to labelled data set. The classifier is then retrained on the updated labelled data set. The steps are then repeated until there are no instances on unlabelled data left. Algorithm 5 demonstrates this procedure.

Algorithm 5: Self-training method

```
Let L be a data set containing labelled instances and U be a data set containing
  unlabelled instances.
Self-training(L, U)
    Create an instance of a classifier C
    while instances left in U do
        Train C on data set L.
        for Instance in U do
            Use C to predict label.
            if Prediction confidence > threshold then
                Add Instance to L.
                Remove Instance from U.
            end
        end
    end
```

4.2.2 Co-training

Co-training is a semi-supervised multi-view learning method (Blum and Mitchell (1998)). The method uses the same main procedural steps as with self-training, but it differs by the fact that it uses two different views to train two different classifiers instead of one view for one classifier. The idea behind co-training being that the different classifiers can help balance out each others mistakes.

In order for the co-training method to work properly it makes the assumptions that the two different views are conditionally independent and that each of the views are separately sufficient for correctly classifying new instances. For real-world classification problems it is not always possible to obtain multiple conditionally independent views. For sleep detection the problem can be solved by using multiple sensors to collect data, each sensor representing a separate view. This is what has been done for our research. However, this solution cannot be used for every classification problem.

As obtaining multiple views is not always possible, some studies have focused on using single-views for co-training. One example being Aridas and Kotsiantis (2015) where two different classifiers, random forest (RF) and Support Vector machines (SVM), are trained on the same view. By splitting the labelled data into training and test sets, and using the split to train the classifiers, the classifier with the highest accuracy is used to make

predictions for the unlabelled data. The instances with the most confident predictions are added to the labelled data and removed from unlabelled. The procedure is repeated until there is no unlabelled data left.

4.3 Multi-view Learning

As previously mentioned, multi-view learning is about machine learning using data represented by several distinct feature sets (Xu et al. (2013)). It is becoming more and more relevant as larger amounts of data can be collected simultaneously from different sources. For instance, a multimedia segment can be described by both audio and video signals. In content-based web-image retrieval, an object is described at the same time by the text surrounding the image and the visual features of the image. These are just some examples, but they still give an indication of the potential of multi-view learning and how it can be applied to a widespread number of issues.

Even when a natural feature split of the data is not possible, manufactured splits can still result in improved performance (Xu et al. (2013)). In our research we have data collected from two separate sensors, so a manufactured split is not necessary. Because of this natural split multi-view learning is therefore a real possibility for obtaining improved performance results during our research.

We have already briefly discussed multi-view learning with regards to semi-supervised learning. However, semi-supervised methods are not the only machine learning methods that are able to use multi-view learning. Supervised learning, among others, is also able to utilize multi-view learning. That being said, more research has currently been conducted on the use of multi-view for semi-supervised learning than supervised learning. Xu et al. (2013) states that one reason for this might be that semi-supervised multi-view learning is often seen as a more general and difficult problem when compared to supervised multi-view learning. Nevertheless, several studies have been conducted on the use of supervised multi-view learning. Wang et al. (2018) proposed using a multi-view learning method for training of an attention-based CNN model. Chen and Sun (2009) introduce a multi-view Fisher discriminant analysis that can be used for both binary and multi-class classification.

In addition to using multi-view learning for supervised learning, using multi-view learning in ensemble learning is also possible. The purpose of ensemble learning is to use achieve better predictive performance by using multiple classifiers/models (Xu et al. (2013)). Training multiple classifiers on different views and then using the combined performance of the classifiers to classify instances could clearly be a possibility for our research. One example of multi-view learning being used for ensemble methods is Xu and Sun (2010) who introduces a embedded multi-view adaption of the well-known Adaboost algorithm called EMV-Adaboost. The results found in the paper validate the effectiveness of the proposed EMV-Adaboost algorithm.

Multi-view learning can also be used for clustering, transfer learning and active learning, among others (Xu et al. (2013)).

4.4 Summary

Arousal (multiclass) classification, semi-supervised learning and multi-view learning are machine learning techniques we believe can potentially improve the performance results for sleep pattern detection. Arousal classification can help provide more detailed information about a subject's sleep pattern while simultaneously help overcome potential limitations found in binary sleep/wake classification. Semi-supervised methods can help reduce the time and resources needed to be invested in obtaining the necessary amount of labelled data used to train and test the models. Multiview learning can help utilize the different sensor views in our data to their fullest extent to potentially improve our performance results. Chapter 6 describes the experiments we conducted during our research that are based on these techniques.

Chapter 5

Methodology

In this chapter we explain how our data was obtained and which equipment was used. A summary of the data sets used in our research is also provided. Next the main procedure is explained and lastly we present more specific information on the data analysis process.

5.1 Data Collection

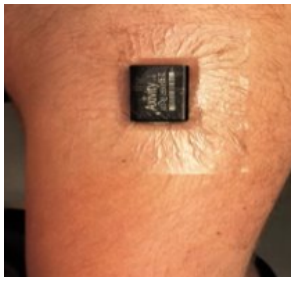
This section presents the process and equipment used to collect the sleep data used in our research. Data was collected for three different data sets. Information about each of these data sets is also provided.

5.1.1 Equipment

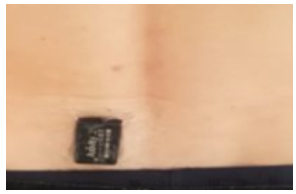
The Axivity AX3 accelerometer[1] can be described as a data logger. The device have dimensions of 23 x 32.5 x 7.6 (mm) and weighs 11g. It contains a high precision accelerometer that is able to detect movement, as well as vibrations and orientation changes for all three axis. The accelerometer samples data at a frequency of 100 Hz. In addition to the accelerometer, the device also contains an on-board memory chip. This chip stores all data detected by accelerometer and markes each stored sample with a precise time-stamp.

For our data collection two such devices/sensors were used. One sensor was placed at the mid-lower back, slightly to the side to avoid discomfort issues, and the second was placed at the front upper right thigh. Figures 5.1 and 5.2 illustrated the exact placement and axis.

[1] https://axivity.com/product/ax3, accessed: 2019-02-03



(a) Thigh sensor placement.



(b) Lower back sensor placement.

Figure 5.1: Sensor placement (figures are printed with permission from the HUNT4 team).

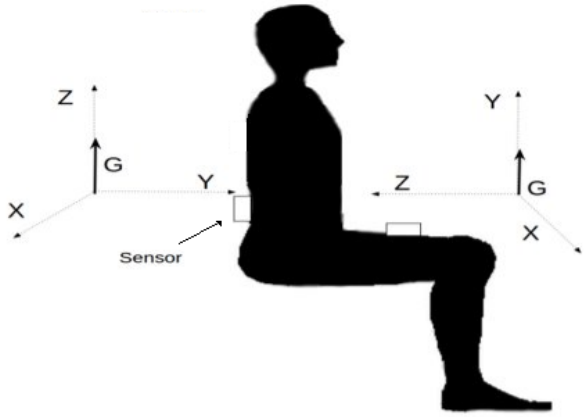


Figure 5.2: Orientation of thigh and back sensors when sitting compared to the gravitational component (figure is printed with permission from the HUNT4 team).

5.1.2 Data sets

When this thesis started no data sets for machine learning experiments were available. As mentioned before, annotations take time and effort. We eventually got access to a data set with recordings of movement, heart rate and PSG. Further experts provided sleep scores for the data set. As this professionally labelled data set was not available during the entire time we also had to initially rely on an artificially labelled data set along with an unlabelled data set. These data sets were created using HUNT 4 recordings and was created for implementing and testing the algorithms we found suitable for our purpose. An overview of these different data sets used during our research is presented as follows.

Artificially Labelled Data

In our previous work (Hay (2018)) no professionally labelled sleep data was made available to us in time for our experiments. The reason behind this is that the data collection and sensor placements is still relatively new, so no data set was available. As a result, we needed to create our own artificially labelled (AL) data set. We accomplished this by obtaining accelerometer data from 9 random participants of the HUNT4 study and manually labelled the data. The subjects were presumed healthy and were of unknown age and gender. Each data point was time-stamped and labelled with a prediction of the current activity of the subject. The activity predictions were provided by a LSTM classifier developed for the HUNT4 study (Hessen and Tessem (2015), Vågeskar (2017), Reinsve (2018)).

For the subject three nights of data was selected. The time frame for a night was set from 10 pm until 10 am. By visualizing the data and using the predicted activities for each data point we manually labelled the data with sleep/wake labels so it could be used in our experiments. It should be noted that we had no prior experience with analyzing and labelling accelerometer data. The exact labelling process used can be described as follows.

If it appeared that a subject had gone to sleep for the night (no/minimal activity could be seen) we located the exact time the subject had started lying down. It is normal to use about 10-20 minutes on falling asleep after going to bed[2]. We therefore used a random number generator to select a number between 10 and 20 as n. The time of falling asleep was then sat as n minutes after the time of lying down.

However, sometimes the data showed meaningful movement activity after lying down which indicated that the subject spent more than 10-20 minutes settling down before going to sleep. If this happened the time of sleep was set to 5-10 minutes after no/minimal movement activity. In addition, if the subject appeared to have woken up during the night the data was labelled as wake from the time of the perceived wake moment (initial major movement) until 0-2 minutes after no/minimal movement from the subject.

Lastly, the time of getting up in the morning was also marked. We focused on the change in predicted activity from lying down to something else to find the exact wake moment. The data was then labelled as wake from the noted activity change time or, if there was significant movement activity before change, from the time of indicated initial major movement. An overview of available artificially labelled subject data can be seen in Figure 5.3.

As professionally labelled sleep data was not available for us in the beginning of this research we initially relied on the AL data set to configure, test and optimize our data analysis process and machine learning methods. This allowed us to implement all pipelines and candidate methods, so once data became available the focus was on model testing and configuration.

[2]https://www.sleep.org/articles/how-long-to-fall-asleep/, accessed: 2019-03-19

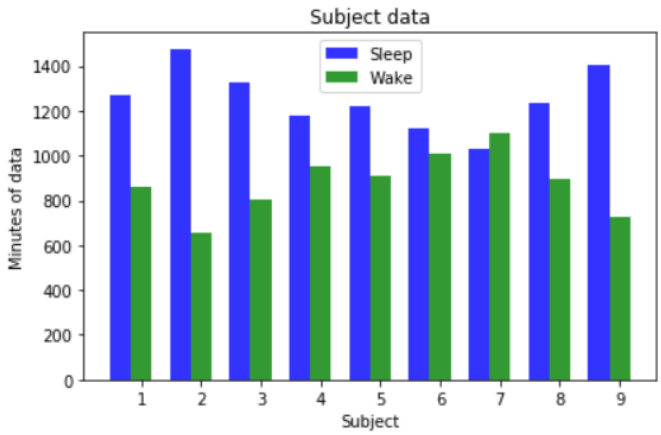


Figure 5.3: Distribution of sleep/wake minutes among the artificially labelled subject data.

Initial Unlabelled Data

For our initial testing of our semi-supervised learning methods an unlabelled data set was needed. We originally used our artificially labelled data set and split it between labelled and unlabelled data. However, we found that this split would potentially leave us with a too small amount of unlabelled data. Therefore, we obtained accelerometer data from 10 new participants of the HUNT4 study to be used as unlabelled data.

Once the data was obtained it was again presumed that all participants were healthy, but of unknown age and gender. The data was of the same format as the AL data set. For each of the participants we selected three nights of data. The time frame for a night was set from 10 pm until 10 am. All data instances were relabelled as 0 for ”Unknown”. In the end, the unlabelled data set consisted of 30 nights (21 900 minutes) of data.

During the initial testing and configuration of our methods one night of data was found inadequate. The inadequacy was the result of it being the first night of data collected from its subject. The sensors had not been worn by the subject in the initial minutes of the selected time frame. As a result the data contained data points that was of no use to us. Knowing we had chosen the first night of data for several other subjects, the possibility remained that the same error had occurred for several other nights of data as well. We therefore decided not to use the unlabelled data set beyond our initial testing and optimizing. As a result, after obtaining professionally labelled data the AL data set (with the labels removed) was used as unlabelled data for validation and testing of our experiments with semi-supervised methods.

Professionally Labelled Data

A professionally labelled (PL) data set was eventually made available to us. The data set mainly consisted of PSG and accelerometer data collected from 19 patients referred to the sleep clinic at St Olavs Hospital, Trondheim, Norway. The subjects were of unknown age and gender and all were diagnosed with some type of sleep disorder. One night of labelled data was made available from each of the subjects. The data was labelled with one of the following labels: Wake, N1, N2, N3, REM, Movement. N1, N2, N3, and REM refers to different sleep labels, and we also assumed the Movement label referred to significant movement during sleep. Consequently, we relabelled these five labels as ”Sleep” and as a result the data only contained two labels (sleep and wake). An overview of the distribution of sleep/wake minutes among the professionally labelled subject data can be seen in Figure 5.4.

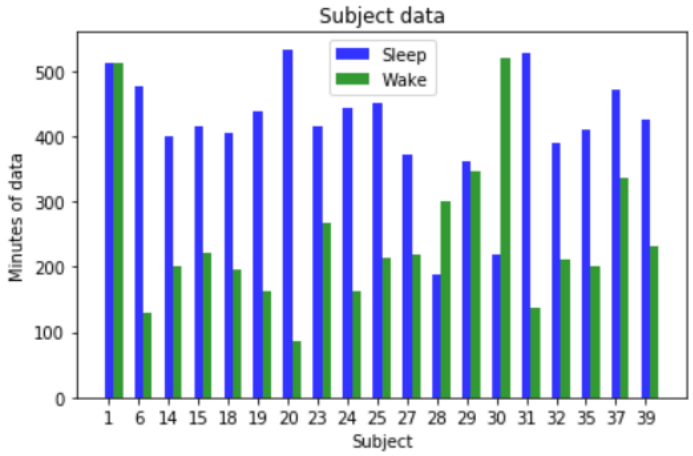


Figure 5.4: Distribution of sleep/wake minutes among the professionally labelled subject data.

5.2 Procedure Overview

An overview of how the raw accelerometer data was analyzed and used for training and testing of classification models can be viewed in Figure 5.5. The first section of the procedure consists of data analysis. Here the data is filtered, segmented and used for feature generation. The result of this section is a feature set which is further split into train and test sets. The second part of the procedure is the model training. This entails using the training set to train a machine learning model. The last step is the classification itself, where the trained model is used to predict the label of all instances in the test set.

In the beginning of our research a significant amount time was used ensure each step

in the entire procedure worked as planned. This has achieved by repeating each step, using the AL data set, and making changes as needed. The unlabelled data set was also utilized if needed.

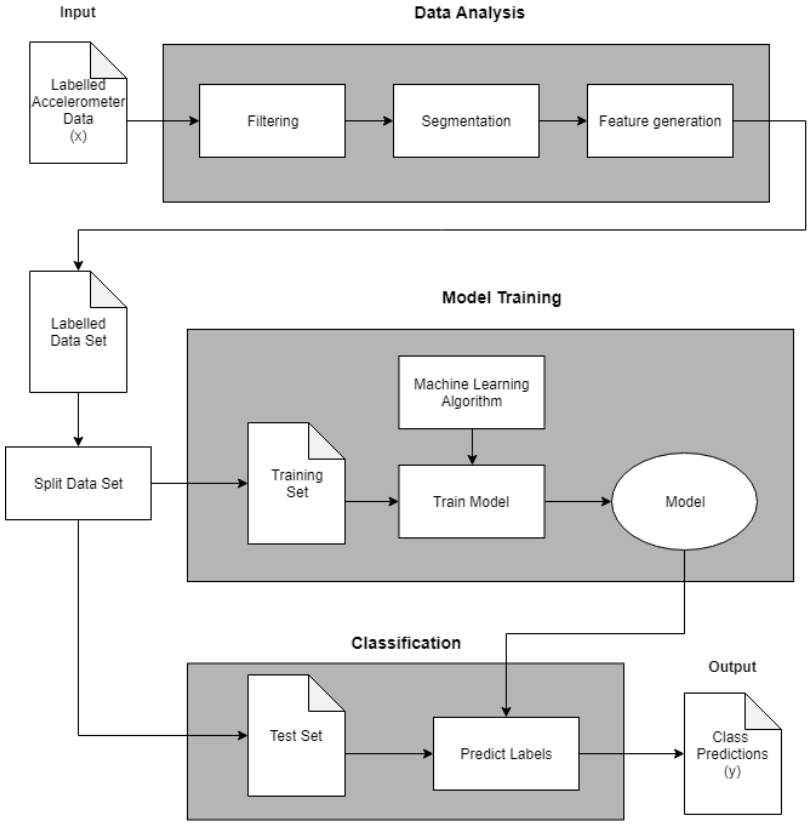


Figure 5.5: Main procedure overview.

5.3 Data Analysis

This section gives a more detailed explanation of how the raw accelerometer data was processed and structured for further use.

5.3.1 Pre-processing

The first step of the pre-processing of the data is removing the time-stamp column and store it separately for potential use later in visualization of the results. The norm of each data point was then added as an additional column. The result is datastream containing the

columns for the x-, y- and z-axis of both sensors, the norm (m), and the predicted label (sleep class). Table 5.6 presents a data sample after pre-processing.

	back_x	back_y	back_z	thigh_x	thigh_y	thigh_z	sleep_class	m
0	-0.735406	-0.036909	0.605948	-0.807736	-0.027052	0.175547	2	1.262277
1	-0.744190	-0.058583	0.624194	-0.633040	-0.353098	-0.102160	2	1.217672
2	-0.738485	-0.128935	0.632936	-0.581218	-0.538292	-0.162988	2	1.271508
3	-0.735936	-0.186155	0.640611	-0.689168	-0.432541	0.087559	2	1.286993
4	-0.764622	-0.203145	0.643296	-1.039125	-0.137490	0.216522	2	1.478280
5	-0.765408	-0.213059	0.644611	-1.180725	0.003846	0.161425	2	1.570654
6	-0.752983	-0.242839	0.643386	-1.084914	0.055377	0.092521	2	1.492839
7	-0.741516	-0.230750	0.634910	-1.011979	0.100317	0.045006	2	1.429123
8	-0.735866	-0.210936	0.613802	-0.895325	0.147737	-0.054772	2	1.337601
9	-0.734471	-0.215570	0.612131	-0.906961	0.239996	-0.123161	2	1.362339

Figure 5.6: Data sample after pre-processing.

As two out of the five papers in section 3.2 focusing on machine learning methods uses filtering in their own pre-processing of data we decided to use filtering as well. As Orellana et al. (2014) did not elaborate on how they performed their filtering, we used the information in Yeo et al. (2017) as basis for our choices for filtering. As a result, we filtered using a fifth-order Butterworth lowpass filter. The sampling rate was set to 10 Hz with the cut-off frequency of the filter at 3 Hz. Figure 5.7 illustrates the datastream for the X-axis of the back sensor before and after filtering was conducted.

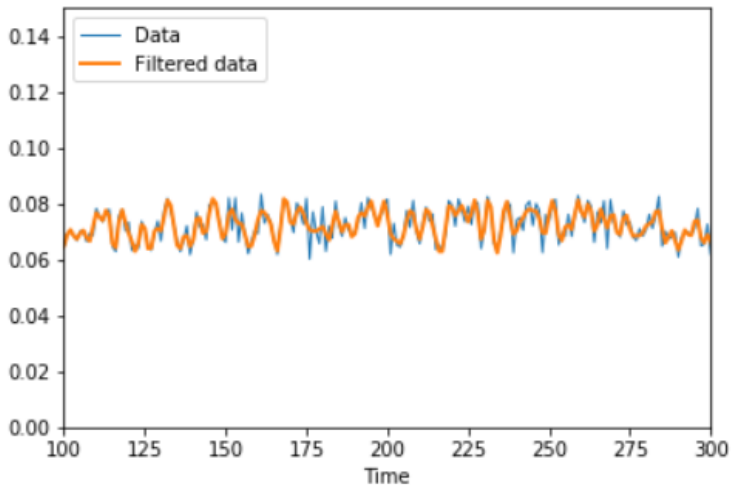


Figure 5.7: Datastream for the X-axis of the back sensor before and after filtering.

5.3.2 Segmentation

For evaluation of a data stream for activity recognition (our activities being sleep and wake) there are typically two main methods of doing so. The first method is to determine the current activity by only focusing on a single data point. However, the information collected from a single data point does not often provide enough data necessary for accurate determination of labels. Therefore, the second method, signal segmentation, is a more commonly used method. It involves segmenting the data stream/signal into epochs of a certain length and then using the information from each epoch to determine the current activity.

When reviewing the papers introduced in chapter 3 it shown that in each paper signal segmentation was utilized as a part of pre-processing of data. With regards to the seven papers focused on machine learning methods five used 30 second epochs while the remaining two used 1 minute epochs. To enable easier comparison to PSG data, which normally also comes in 30 second epochs, and because of its clear common use, we also decided to use 30 second epochs for our own signal segmentation.

It should be pointed out that after segmentation if an epoch contained an equal split between the amounts of sleep and wake instances the epoch would be labelled as wake, otherwise the epoch label would always be the majority instance.

Sliding Window Method

A common method for signal segmentation, that we also used, is the sliding window method. When using this method, after completing the evaluation of one epoch, the window slides along the signal after evaluating one epoch to encompass and evaluate the next epoch. Two aspects are important to take into consideration when applying this method: the overlap between adjacent windows and the length of the window itself.

With regards to the window length, we discovered based on related work that it is quite common to have the window encompass more than just the target epoch. As seen in more earlier literature on sleep detection, such as Sadeh et al. (1994) and Cole et al. (1992), the authors often use information from previous and following epochs when trying to determine the sleep/wake prediction of a specific epoch. More specifically, based on the content of the papers summarized in Section 3.2, the most common window used is a sliding window with a length of 10-11 minutes. The window also normally has a center epoch of 30 seconds that would be considered the target epoch.

For that reason, we selected to use a sliding window consisting of 21 epochs (10,5 minutes) with a center of a 30 second target epoch. Using a window of this size leads to a significant overlap between the adjacent windows, about 95%. To be more specific, 20 out of the 21 epochs found in a window representing a target epoch would also be found in the following window. Figure 5.8 shows an illustration of the sliding method used.

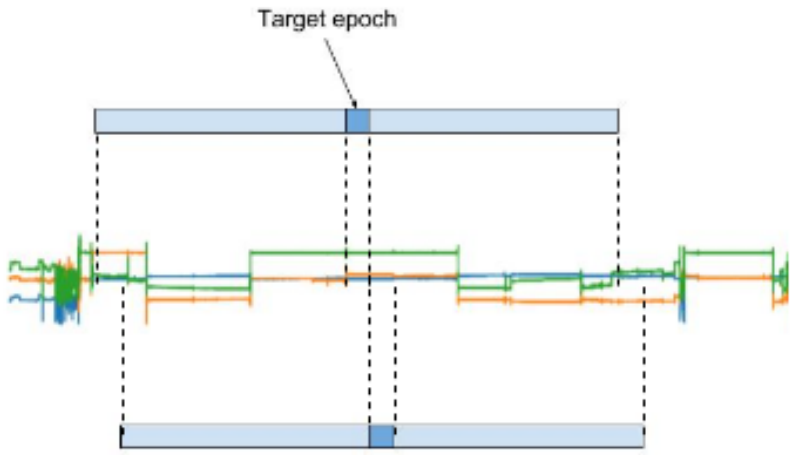


Figure 5.8: Illustration of the sliding window used for segmentation.

Using a sliding window consisting of multiple epochs when evaluating sleep data makes sense. In order to accurately determine if a person is asleep or awake during a specific epoch it is often necessary to also view the previous and following epochs. A single epoch can prove to be misleading. For example, if looking at movement data from when a person is simply changing position while sleeping it could give an indication that the person is instead awake. However, if it is also known that there has been minimal movement leading up to and following that epoch the data more correctly indicates that the person is actually asleep and simply changing position. Looking at the surrounding epochs as well as the target epoch could therefore prove to give a more accurate assessment.

5.3.3 Feature Selection

Our feature selection were a combination of the features used in our previous work with sleep detection (Hay (2018)) and some additional features added by us during this research. Each feature was calculated for the x-, y-, and z-axis of both sensors and the norm of each data point. Unless otherwise specified, the features are calculated for the entire sliding window of 21 epochs. We ended up with a total of 100 features and the final selection can be seen in Table 5.1

Feature	Equation	Description
Mean	$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \quad (5.1)$$	Mean value of the accelerometer data. N is the data length and x is the accelerometer data.
Root mean square	$$rms(x) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} \quad (5.2)$$	Square root value of the mean square of the accelerometer data.
Standard deviation	$$std(x) = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N} (x_i - \overline{x})^2} \quad (5.3)$$	Standard deviation of the accelerometer data. A measure used to quantify the amount of variation or dispersion of a set of data values.
Kurtosis	$$Krt(x) = \frac{E[(x-\overline{x})^4]}{std(x)^2} - 3 \quad (5.4)$$	Kurtosis of the accelerometer data. A measure of the peakedness of the data.
Skewness	$$Sk(x) = E\left\|\left(\frac{x-\overline{x}}{std(x)}\right)^3\right\| \quad (5.5)$$	The skewness of the distribution of the accelerometer data. A measure of the assymmetry of the probability distribution.
Sum of values	$$sum(x) = \sum_{n=1}^{N} x_i \quad (5.6)$$	Sum of values of the accelerometer data.
Coefficient of variation	$$CV(x) = \frac{std(x)}{mean(x)} \quad (5.7)$$	The ratio of the standard deviation to the mean of the accelerometer data.
Zero crossings	$$ZCR(x) = \left\|\{\|i \in N\|(2 \leq i \leq N) \wedge (x_i \cdot x_{i-1} < 0)\}\right\| \quad (5.8)$$	The number of zero crossings in the accelerometer data.

Interquartile range	$$IQR(x) = b(x) - a(x) \quad (5.9)$$	IQR is the difference between the 75th and 25th percentile of the accelerometer data. A measure of the dispersion. b and a represents the 75th and 25th percentile respectively.
Min-max-mean	$$min\_max\_mean(x) = \frac{1}{K}\sum_{k=1}^{K} c_k - d_k \quad (5.10)$$	The average of the differences between local minimums and maximuma in the accelerometer data. K is the number of local maximums/minimums, and c and d represents lists of local maximums and minimums respectively.
Energy	$$E_x = \sqrt{\sum_{i=1}^{N}(x_i - \mu)^2} \quad (5.11)$$ $$Energy = \frac{1}{3N}(E_x + E_y + E_z) \quad (5.12)$$	The signal's energy. x_i is the value at the i position on the x-axis. μ is the mean value of the signal. N is the length of the signal and E_x, E_y, and E_z is the energy of the x-, y- and, z-axis, respectively.
Number of maximums - central epoch		Number of local maximums in the accelerometer data for the central epoch.
Number of maximums - first 10 epochs		Number of local maximums in the accelerometer data for the combined first 10 epochs.

Number of maximums - last 10 epochs		Number of local maximums in the accelerometer data for the combined last 10 epochs

Table 5.1: Final feature selection.

Chapter 6

Experiments

In this chapter we explain the experimental set up and what the experimental procedure was for each experiment.

6.1 Software Libraries

Scikit-learn[1] is a free machine learning software library made for the Pyhton programming language. It is build on Python's own numerical and scientific libraries (NumPy and SciPy) and it contains simple and effective tools for classification and regression, along with other methods for data analysis and data mining. For our experiments we used scikit-learns classification algorithms for decision tree and random forest.

XGBoost is not available as an algorithm in scikit-learn. Intead the algorithm is available in its own software library. Therefore, for experiments with XGBoost we simply used the XGBoost library's own XGBoost classifier algorithm.

6.2 Main Set-up

For the selection of parameters for our three main classifiers we selected them based on our previous work (Hay (2018)). The parameters for the decision tree classifier was set to default. For random forest we set the number of trees/estimators parameter to 20 and all other parameters to the default setting. For XGB we decided to use 150 estimators, and any other parameters for the classifier where set to default.

For training and testing purposes our PL data set was divided into training and test sets. The test set consisted of data from two randomly selected subjects. The data from the

[1] https://scikit-learn.org/stable/, accessed: 2019-05-08

remaining 17 subjects was assigned as training set which was further divided into training and validation sets, with another two randomly selected subjects assigned as validation. The subject IDs selected for the test set was number 14 and 18 and the subject IDs selected for the validation set was number 32 and 35. The remaining subjects made up the training data.

The training set was used to train the model, while the validation set was used to give an evaluation of the model while the parameters were being tuned. The test set was used last to provide an unbiased evaluation of the final model. For this final testing the validation set was added as a part of the training set.

6.3 Supervised Learning Classifiers

In our previous work with sleep/wake classification (Hay (2018)) we compared the performance of five different supervised learning classifiers: decision tree (DT), random forest (RF), articifial neural network (ANN), Gaussian naive Bayes (GNB), and extreme gradient boosting (XGB). All of these classifiers were trained using the features listed in Table 5.1, with the exception of all energy related features. The results of the work showed that XGB, RF, and DT had the best performance results. Based on this we decided to focus on the continued use of DT, RF, and XGB classifiers during this research.

Therefore, in the beginning of our experiments we trained and tested each of these three classifiers using our PL data set to create a performance baseline for sleep/wake classification that we would attempt to improve on. The classifiers were trained using the same parameters used in our previous work (see section 6.2).

To also obtain an insight into our feature selection we decided to also train and test XGBoost while using different feature selections. For this purpose we applied the same feature selection algorithms that we had experimented with in our previous work (Hay (2018)). The three main types of feature selection were [2]: removing features with low variance, univariate feature selection and feature selection using a meta-transformer that utilizes importance weights for selecting features (SelectFromModel).

6.3.1 Feature Selection Set-up

For our experiment with feature selection we decided to utilize scikit-learn's feature selection algorithms. The description and set-up for each of the algorithms is described as follows.

[2] https://scikit-learn.org/stable/modules/feature_selection.html, accessed: 2019-05-07

Removing Features with Low Variance

The *VarianceThreshold* method is a feature selection method where all features with a variance below a set threshold is removed. For our purposes we used a threshold of 80%. As a result, 35 features were removed from the feature set and 65 features remained.

Univariate Feature Selection

The univariate feature selection method uses univariate statistical tests when choosing the best features. We decided on using the *SelectKBest* method, which only keeps the k highest scoring features. k was set equal to 20 and we used *f_regression*[3] as the scoring function. As a result, 80 features were removed from the feature set and 20 features remained.

Feature Selection using SelectFromModel

The *SelectFromModel* method is meta-transformer used for feature selection. It is used along with an estimator that calculates the *coef_* or *feature_importances_* values used to determine if a feature should be removed or not. We used two different estimators for feature selection with *SelectFromModel*: *L1-based* and *Tree-based*.

We set the penalty parameter C to be 0.01 for the *L1-based* estimator and as a result 41 features were removed from the feature set and 59 features remained. With regards to the *Tree-based* estimator we decided to set the number of estimators to 50. This resulted in 72 features being removed and 28 features remaining.

6.3.2 Results

Based on the results of the experiments we decided not to use any feature selection methods in our remaining experiments for binary sleep/wake classification and instead use all features as shown in Table 5.1. The specific performance results for XGBoost when using feature selection can be seen in chapter 7.

6.4 Multiclass Classification

Evaluating the impact that the use of multiclass classification has on the performance of sleep pattern detection is one of our research questions:

- **RQ2**: How does multiclass classification affect the overall performance results for sleep pattern detection?

To evaluate RQ2 we first needed to determine which class(es) should be added to our classification problem. We have already discussed in chapter 4 the impact arousals have on

[3] See footnote 2.

the performance of binary sleep/wake classification. In addition, arousal annotations were made available to us for the PL data set. Based on this, we decided to select ”Arousal” as an additional label/class. This would extend the binary sleep/wake classification problem into a multiclass classification problem with three distinct classes: sleep, wake, and arousal.

6.4.1 Procedure

Something that should be noted is that the arousal annotations that accompanied our PL data set was made automatically and not manually. This means that the annotations could only be relied on to a certain point and it was very likely that the annotations contain mistakes.

Relabelling of Data

Before any experiments with multiclass classification are carried our additional pre-processing of the data should be conducted. The already pre-processed PL data set was only labelled with sleep and wake labels and for our multiclass experiments we would need the data to contain arousal labels as well. To accomplish this we manually relabelled the data using the available arousal annotations that accompanied the raw data.

The arousal annotations came for each night of data gathered for a subject and contained information about each arousal occurrence. For each occurrence the start and end time of the arousal was noted. In addition, the arousal type was also specified. Based on this information we manually relabelled our PL data set. An overview of all types found in the annotations can be seen in Appendix B. The manual relabelling can be described as follows. Based on the available arousal annotations, for each occurrence of an arousal we noted which 30 second epoch(s) (timestamp(s)) it took place in. Following this, we then matched the timestamps from our already pre-processed PL data set with this list of arousal timestamps. If an epoch from the PL data set contained one or more arousals then it would be relabelled as ’arousal’ (we used 0 to represent this label). Overviews of the available subject data after relabelling can be seen in Figures 6.1 and 6.2.

After relabelling had been conducted DT, RF, and XGB classifiers were trained and tested using the relabelled data.

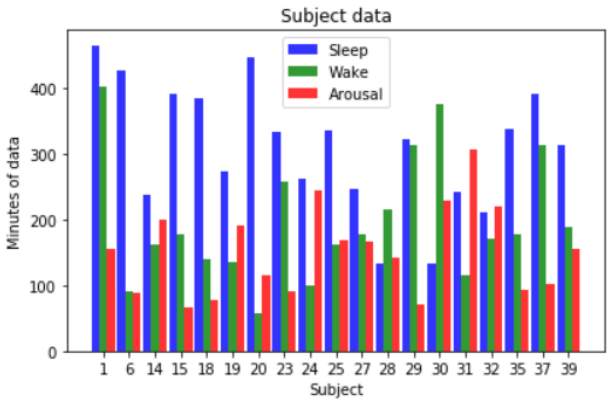


Figure 6.1: Distribution of sleep/wake/arousal minutes among the PL subject data with all arousal types.

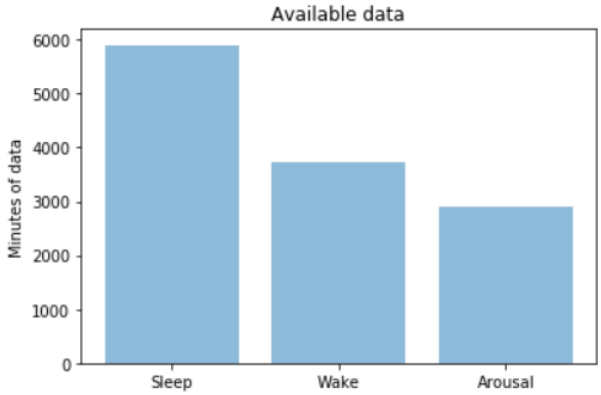


Figure 6.2: Total amount of data available in the PL data set with all arousal types.

6.4.2 Feature Selection

Previously we decided not to use any feature selection algorithms for our binary classification. However, we still needed to obtain insight into the suitability of features for multiclass classification. We therefore decided to once again train and test XGB while using the same feature selection algorithms and parameters as what has been described previously in section 6.3.1. However, the number of features removed differs for multiclass classification.

When using *VarianceThreshold* 35 features were removed and 65 remained. For *SelectKBest* 80 features were removed and 20 remained. With regards to *L1-based* and *Tree-based* feature selection, 31 and 66 features were removed and 69 and 34 features remained, respectively.

We did not use any feature selection algorithms for our remaining experiments with multiclass classification.

Selective Relabelling - PLM

As mentioned, one of the problems with the received arousal annotations was that they were all made automatically and not manually. This reduced the confidence of the annotations. To potentially overcome this issue we decided to conduct selective relabelling using only one specific type of arousal that we had higher confidence in, namely periodic limb movement (PLM) arousals.

PLMs in sleep can be described as short (0.5- to 5.0-second) lower-extremity movements, which normally occur during sleep at 20- to 40- second intervals. Evidence suggests that PLMs are an indication of instability in sleep and they can be seen in various sleep disorders (Picchietti and Winkelman (2005)).

During the PSG recording sessions for the subjects foot movement would have been measured by an electrode placed on the shin to give, among other things, an indication of when PLM arousals occured. There should therefore be a certain level of accuracy for the occurence of the PLM arousals. In addition, because one of our sensors had been placed on the upper thigh it would be likely that the movement patterns of the foot had been picked up by the sensor and should therefore be represented in our data. Based on this reasoning we decided to attempt selective relabelling where only PLM arousals were noted.

The selective relabelling process was similar to the general relabelling process descibed previously. The one exception being that we only noted the occurence of PLM arousals and ignored all other types. Any epoch containing an arousal not of the PLM type would keep its original sleep or wake label. The overviews of the available subject data after selective relabelling for PLM arousals can be seen in Figures 6.3 and 6.4.

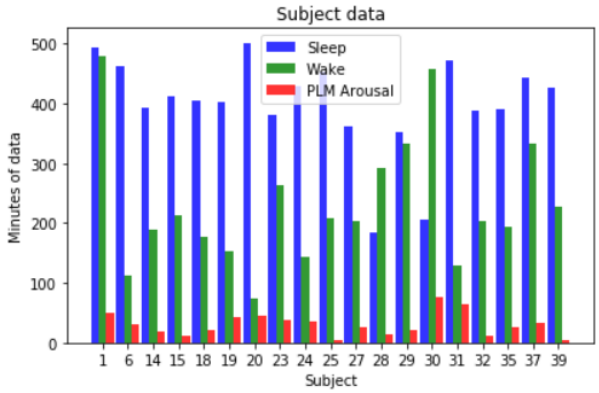


Figure 6.3: Distribution of sleep/wake/arousal minutes among the PL subject data with only PLM arousals.

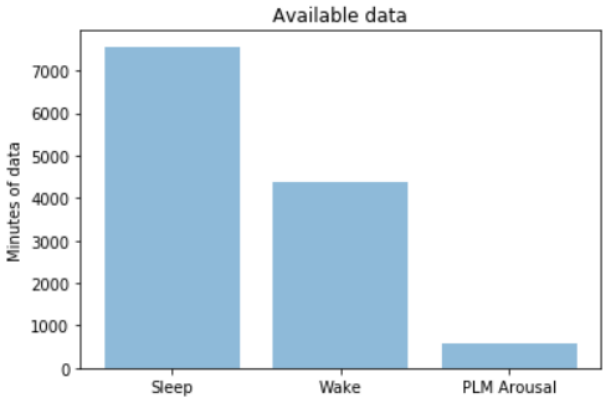


Figure 6.4: Total amount of data available in the PL data set with only PLM arousals.

After selective relabelling had been conducted DT, RF, and XGB classifiers were trained and tested using the relabelled data.

Balancing Dataset - PLM

As seen in Figure 6.4 the resulting data set after selective relabelling for PLM is quite unbalanced, especially with regards to PLM arousal instances. Having such an imbalance can lead to the classifiers potentially overlooking the arousal class completely and simply labelling instances as sleep or wake as it would result in the highest accuracy. To try and overcome this issue we decided to balance the data set to obtain an more even distribution of labels.

Before training the classifiers we duplicated the original arousal instances in the training set four times. This gave us a training set that contained fives times the amount of arousal instances. After this duplication process DT, RF, and XGB classifers were trained and tested once more.

6.4.3 Performance Evaluation

The evaluation metrics as stated in section 2.7 is only meant for binary classification. As a consequence we had to update the metrics for our multiclass experiments. The confusion matrix, along with accuracy, could still be used for evaluating the performance but with updated definitions. In addition, we also decided to use precision and recall metrics. Precision, with regards to a specific class, can be defined as the percentage of instances labelled as that class that was actually correct. Recall, with regards to a specific class, can be defined as the percentage of the class' instances that was labelled correctly. The updated confusion matrix for our multiclass experiments can be seen in Table 6.1. The updated

definition for accuracy is also presented below along with the metrics for precision and recall.

Actual Class	**Predicted Class**		
	Arousal	*Sleep*	*Wake*
Arousal	AA	AS	AW
Sleep	SA	SS	SW
Wake	WA	WS	WW

Table 6.1: Confusion Matrix - Multiclass classification.

Total accuracy rate:

$$Acc = \frac{\text{AA + SS + WW}}{\text{AA + AS + AW + SA + SS + SW + WA + WS + WW}} \tag{6.1}$$

Arousal Class:

$$Precision = \frac{\text{AA}}{\text{AA + SA + WA}} \tag{6.2}$$

$$Recall = \frac{\text{AA}}{\text{AA + AS + AW}} \tag{6.3}$$

Sleep Class:

$$Precision = \frac{\text{SS}}{\text{SS + AS + WS}} \tag{6.4}$$

$$Recall = \frac{\text{SS}}{\text{SS + SA + SW}} \tag{6.5}$$

Wake Class:

$$Precision = \frac{\text{WW}}{\text{WW + AW + SW}} \tag{6.6}$$

$$Recall = \frac{\text{SS}}{\text{WW + WA + WS}} \tag{6.7}$$

6.5 Co-training with Single-view

After completing the multiclass experiments we moved our focus to our final research question:

- **RQ3**: How do ensemble methods affect the overall performance results for sleep pattern detection?

To start of we decided to evaluate the use of semi-supervised ensemble methods for sleep pattern detection. Aridas and Kotsiantis (2015) presents a co-training method with

single-view combining random forest (RF) and support vector machines (SVM). Despite Aridas and Kotsiantis (2015) not using the method described for sleep detection we found this combination for a semi-supervised machine learning method to be an interesting and potentially promising possibility, especially as RF has already shown promising result during our own previous research.

However, we were unsure of the SVM algorithms suitability for sleep/wake classification. In the worst case scenario, when dealing with highly imbalanced data, an SVM typically requires $O((N_p + N_n)^3)$ time for training (Tang et al. (2009)), with N_p and N_n representing positive and negative samples, respectively. Consequently, since we are using an unbalanced data set in this research we decided to replace the SVM algorithm to avoid any potential run-time issues and test the method described with RF and XGB instead.

6.5.1 Procedure

As mentioned, we decided to use RF and XGB classifiers for our co-training with single-view (CoSV) method. The procedure can be described as follows.

Each of these classifiers is first trained and tested using the hold out method on the initial labelled data set. The most accurate classifier of the two is then selected and used to predict the labels for the instances of unlabelled data. Based on the prediction confidence of the classifier, an unlabelled data instance is added to the labelled data and removed from unlabelled if its confidence is above a set threshold k. This is repeated, using the updated data sets, until there is no unlabelled data left and all instances is labelled. To evaluate how well the semi-supervised method performed the resulting data set of labelled data is lastly used to train DT, RF, and XGB classifiers. The classifiers are then used to label the test cases in our standard test set (see section 6.2).

The pseudocode for the altered algorithm we used in our experiments is shown in Algorithm 6 and an overview of the procedure can be viewed in Figure 6.5.

We set the value of the threshold mentioned in the algorithm do be the highest confidence value found minus a set value k. Initially k was set to 0.001. To avoid stagnation and run-time issues we doubled k every time less than a thousand unlabelled instances was added to the labelled data set until k reached a maximum value of 0.64

Algorithm 6: Co-training with single-view algorithm (Aridas and Kotsiantis (2015))

Let L be a data set containing labelled instances.
Let U be a data set containing unlabelled instances.
`CoSingleView(`L, U`)`
  Create an instance of a Random Forest (RF) classifier.
  Create an instance of an Extreme Gradient Boosting (XGB) classifier.
  while *instances left in* U **do**
    Using the hold out method (25%) split L into train/test sets.
    Compare the performance of the RF and XGB classifiers while using the splits.
    Select the classifier with the highest accuracy as C.
    Train C on the entire data set L.
    Use C to predict classes for U.
    for *Instance in* U **do**
      if *Prediction confidence* $>$ *threshold* **then**
        Add Instance to L.
        Remove Instance from U.
      end
    end
  end
  Return L.

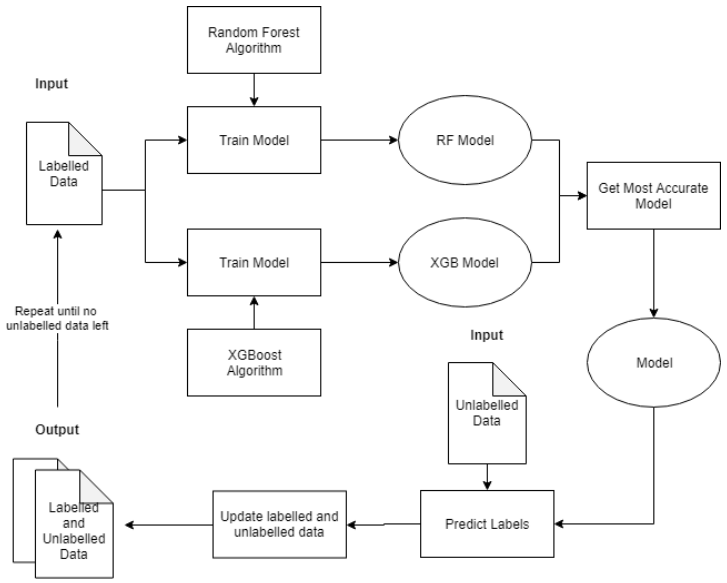


Figure 6.5: Procedure overview for the co-training with single-view method.

6.5.2 Selection of Initial Labelled Data

During our first experiment with CoSV we wanted an accurate evaluation of how the method worked when starting with a small enough labelled data set. As a consequence, we decided to select the data from only five randomly selected subjects in our standard training set (see section 6.2) as use it as the initial labelled data.

The subject IDs selected for the labelled data set were 1, 23, 29, 37, and 39. The data from the remaining 12 subjects in the training set was set as unlabelled data along with all the data from the AL data set. After using the CoSV method to relabel all instances of unlabelled data the resulting labelled data set was evaluated using DT, RF, and XGB classifiers and afterwards we moved on to test another initial selection of labelled data.

The second selection we tested consisted of choosing the entire training set as the initial selection of labelled data, and not just data from 5 out of 17 subjects. In this case, only the AL data set was used as unlabelled data. The reasoning behind this test was to see if obtaining a bigger data set for our experiments, based on the entirety of the labelled data we already had, would help improve the performance. Once again, after using the CoSV method to relabel the unlabelled data, the resulting labelled data set was used as training data to train DT, RF, and XGB classifiers and our standard test set was used for testing.

6.6 Co-training with Multi-view

After the completion of the CoSV experiments we moved on to testing the use of another semi-supervised method. As discussed in chapter 4, since we have data simultaneously gathered from two sensors it is possible for us to use multiview learning. Based on this knowledge, we created a semi-supervised method utilizing multiple views: a co-training with multi-view (CoMV) method.

6.6.1 Procedure

For the different views we decided to use the thigh view and back view. The thigh and back views consisted of all calculated features connected to the thigh and back sensors, respectively. A classifier is trained on each view and used to make predictions on the unlabelled data. Based on the combined prediction confidences of the classifiers, an unlabelled data instance is added to the labelled data and removed from unlabelled if its confidence was above a set threshold k. This is repeated until there is no unlabelled data left. For the selection of classifier we experimented using DT, RF, and XGB classifiers separately. To evaluate how well the semi-supervised method performed the resulting data set of labelled data is lastly used to train DT, RF, and XGB classifiers. The classifiers are then used to label the test cases in our standard test set.

The pseudocode for the algorithm we used in our experiments for CoMV is shown in Algorithm 7 and an overview of the procedure can be viewed in Figure 6.6.

Algorithm 7: Co-training with Multi-view method.

```
Let L be the data set containing all labelled instances.
Let L1 be a data set containing labelled instances with view 1 (thigh view).
Let L2 be a data set containing labelled instances with view 2 (back view).
Let U be a data set containing unlabelled instances.
CoMultiView(L, U)
    Create two classifier instances C1 and C2 using the same classification
     algorithm.
    while instances left in U do
        Train C1 using L1 and C2 using L2.
        for Instance in U do
            Use C1 and C2 to predict class.
            Multiply the prediction confidences from each classifier into one
             confidence.
            if Confidence > threshold then
                Add Instance to L, L1 and L2.
                Remove Instance from U.
            end
        end
    end
    Return L.
```

We set the value of the threshold to be the highest confidence value found minus a set value k. Initially k was set to 0.001. To avoid stagnation and run-time issues we doubled k every time less than a thousand unlabelled instances was added to to the labelled data set until k reached a maximum value of 0.64.

For the selection of the initial labelled data sets we used the same selections as we decided on for the previous method (CoSV). To properly evaluate the method we also used the resulting data sets, along with the test set, to train and test DT, RF, and XGB classifiers.

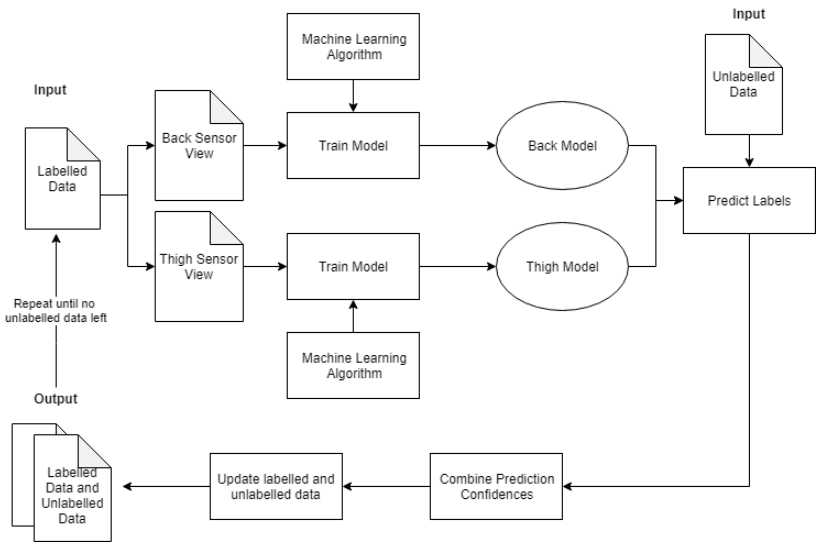


Figure 6.6: Procedure overview for the co-training with multi-view method.

6.7 Supervised Multi-view Learning

After conducting our experiments with semi-supervised methods we wanted to to test out another way to use ensemble methods together with multi-view learning. As we mentioned in chapter 4, supervised learning methods are also able to be used for multi-view learning. We therefore decide to create a supervised multi-view (SuMV) method that utilizes multiple classifiers and multiple views.

6.7.1 Procedure

We selected the thigh view, back view and the combined view as our view selection. The thigh and back view consisted of all calculated features connected to the thigh and back sensors, respectively. The combined view consisted of all features as described in section 5.3.3. A classifier is trained on each of the different views and a majority voting scheme would be used to determine the final class predictions. For the selection of classifier we first experimented using DT, RF, and XGB classifiers separately.

The pseudocode for the algorithm we used in our experiments for supervised multi-view is shown in Algorithm 8 and an overview of the procedure can be viewed in Figure 6.7.

Lastly, we also tested using the three different classifiers together (DT, RF, and XGB). In that case the algorithm differed slightly from what is shown the Algorithm 8. Instead of only creating three instances for one type of classifier, three instances would be created

Algorithm 8: Supervised Multi-view Algorithm

Let L be a data set containing labelled instances.
`SupervisedMultiView(`L`)`
  Create three classifier instances $C1$, $C2$ and $C3$ using the same classification algorithm.
  Split L into a training set TL and a test set CL.
  Split the features of TL and CL into three views (back, thigh and combined view).
  Train the classifiers, $C1$, $C2$ and $C3$, on each of the three views.
  for *Instance in CL* **do**
    Use $C1$, $C2$ and $C3$ to predict class.
    Use majority voting to determine the class of the instance.
  end

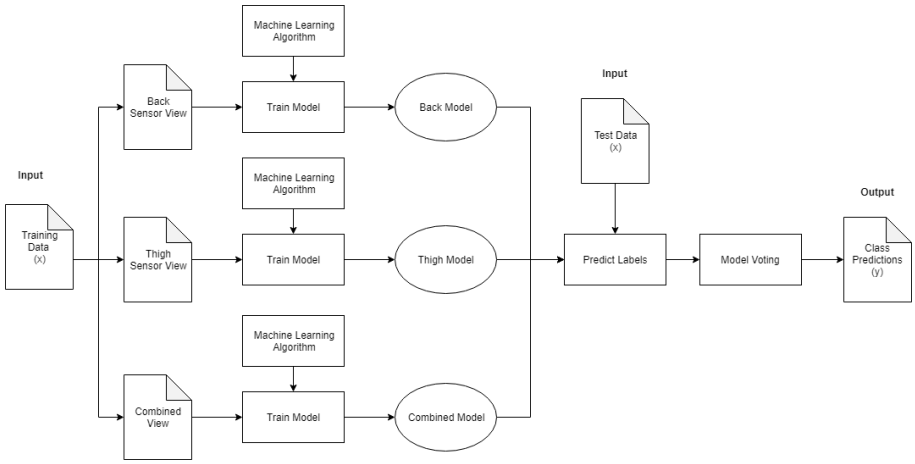


Figure 6.7: Procedure overview for the supervised multi-view method.

for each of the three types of classifiers. A classifier instance of each type would then be trained on each view and once again majority voting be used to determine the class of an instance.

6.8 Supervised Multi-view with Clustering

Because we utilize a majority voting scheme in the supervised multi-view (SuMV) method it would be possible to use the percentage that voted for the final label as a prediction confidence of sorts. Knowing this, we decided to attempt to change/improve the prediction of instances with a low prediction confidence.

The way we decided to attempt his was to first have all test instances labelled using

the SuMV method. Then, using the prediction confidences from SuMV, all instances with a low enough confidence would be relabelled using a combination of clustering and the kNN method.

6.8.1 Set-up

For the kNN classifiers we set the parameter k to 10 and we used the kNN classification algorithm from scikit-learn. With regards to the clustering algorithms we also used the respective algorithms found in scikit-learn.

6.8.2 Procedure

The pseudocode for the basic algorithm we used in our experiment is shown in Algorithm 9 and the procedure can be described as follows.

First we decided to alter the SuMV method by having it also return the prediction confidences along with the predictions. Following this we separated all test instances into two groups: group A and group B. Group A consisted of instances with a prediction confidence above a set threshold. This group keeps their predicted labels. Group B then consists of instances with a prediction confidence below a set threshold and all instances in this group would be relabelled. The next step was then to use group A to train a kNN classifier. Group B was then separated into n clusters using a clustering algorithm. Selecting a representative from each cluster, the kNN classifier is then used to label each representative and all instances in each cluster receives the same label as their representative. After this relabelling group A and group B would rejoin and we have the final predictions.

For the selection of classifiers for the SuMV part of the experiment we first used DT, RF, and XGB classifiers separately and then finally all together. When used together we ended up with 9 classifiers for the majority voting. We set the value of the threshold to be 0.8. In other words, any instance with a prediction that less than 80% of the classifiers voted for was relabelled using clustering. For example, when only using DT classifiers for an instance not to be relabelled all three created classifiers would have to have voted for the final prediction of that instance. Two votes would not be enough as 2/3 is below 0.8. For the clustering algorithm we decided to experiment with both k-Means clustering and agglomerative hierarchical clustering.

Selection of Number of Clusters

When using k-means clustering we initially split the data from group B into two clusters: one for sleep and one for wake. This resulted in poor performance scores, which was not a complete surprise as the k-Means algorithm in scikit-learn is a general-purpose method and

Algorithm 9: Supervised Multi-view with Clustering Algorithm

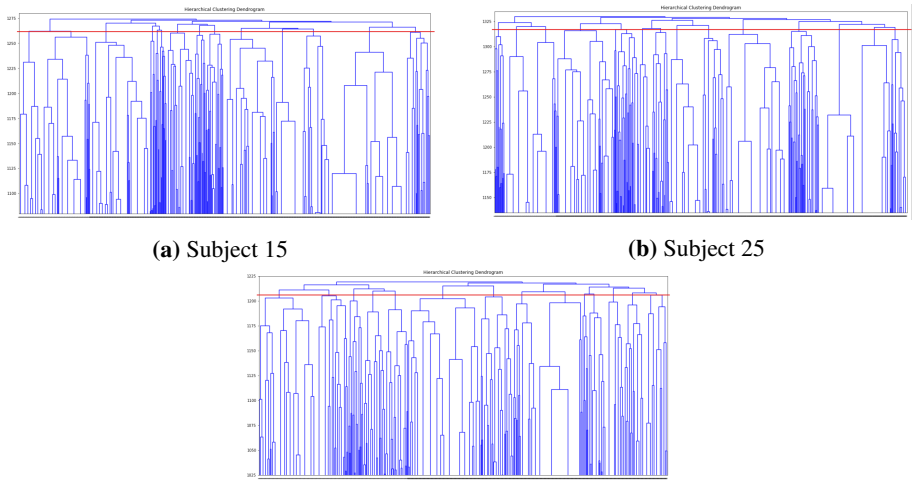
Let L be a data set containing labelled instances.
`SupervisedMultiViewClustering(`L`)`
Create three classifier instances $C1$, $C2$ and $C3$ using the same classification algorithm..
Split L into a training set TL and a test set CL.
Split the features of TL and CL into three views (back, thigh and combined view).
Train the classifiers, $C1$, $C2$ and $C3$, on each of the three views.
Create two empty data sets A and B.
for *Instance in CL* **do**
    Use $C1$, $C2$ and $C3$ to predict class.
    Use majority voting to determine the class of the instance.
    Set the majority voting percentage as the prediction confidence.
    if *Prediction confidence>threshold* **then**
        Add instance with predicted class to A.
    else
        Add instance without predicted class to B.
end
Create an instance of a k-NN classifier.
Train the k-NN classifier on A.
Use a clustering algorithm to split B into n clusters.
for *cluster in clusters* **do**
    Select an instance I from the cluster as the cluster representative.
    Use the k-NN classifier to predict the class of I.
    Label the entire cluster with the predicted class.
end
Combine A and B.

might not do well with clusters that have a specific shape or where the standard euclidean distance is not the right metric[4].

To find a more appropriate number of clusters for the k-means algorithm we decided to use hierarchical clustering to create a dendrogram of one night of data for three randomly selected subjects. We then drew a line at approximately the same distance from the top of the dendrograms and counted the number of clusters it intersected with. The created dendrograms can be seen in 6.8. The number of clusters we got was 14, 15 and 15. As a result we decided on using 15 as the number of clusters for both k-Means and agglomerative hierarchical clustering.

To confirm our choice we also performed supervised multi-view with K-means clustering using DT classifiers while using various numbers of clusters. To avoid potential

[4]https://scikit-learn.org/stable/modules/clustering.html, accessed: 2019-03-26



(a) Subject 15

(b) Subject 25

(c) Subject 35

Figure 6.8: Dendrograms

overfitting, the training and validation sets was used during this testing. The results can be seen in Table 6.2. The table shows that using 35 clusters had the highest overall accuracy but also that using 15 clusters resulting in the highest performance scores for both specificity and G-mean. We can easily disregard the highest sensitivity score (using 5 clusters) as it also comes with the lowest specificity score. Based on these results we found that our choice of using 15 clusters was valid and reasonable, especially as we were interested in the combined best performance for both sensitivity and specificity,

SuMV with K-Means Number of Clusters	Accuracy	Sensitivity	Specificity	G-Mean
5	0.8768	**0.9938**	0.6501	0.8038
10	0.9097	0.9913	0.7518	0.8633
15	0.9031	0.9650	**0.7833**	**0.8694**
20	0.9064	0.9863	0.7518	0.8611
25	0.9089	0.9900	0.7518	0.8627
30	0.9097	0.9850	0.7639	0.8674
35	**0.9110**	0.9875	0.7627	0.8679
40	0.9105	0.9869	0.7627	0.8679

Table 6.2: Performance results - Supervised Multi-view with K-means clustering for DT

Selection of Cluster Representative

As a part of the pseudo-code for our method as described in Algorithm 9 one of the final steps involves choosing a cluster representative to classify and them relabel its entire clus-

ter. When using k-means clustering there was already an easy solution as each cluster had an instance already labelled as the centroid of the cluster (see section 2.6.1). As a result, when we conducted our experiments with SuMV with K-means we used these centroids as the cluster representatives.

However, agglomerative hierarchical clustering does not use centroids when creating clusters. We therefore had to find a different cluster representative. We briefly considered randomly selecting an instance from each cluster and use them as the representatives. However, we quickly discarded this idea as the instances could easily turn out to be potentially bad representations of their clusters. Instead, we decided to calculate the mean value of all instances found in each cluster and use the result as the cluster representatives.

6.9 Final Experiment

After the completion of the experiments described above we wanted a final evaluation of our best performing method. As mentioned in section 5.1.2 our PL data set consists of data collected from individuals diagnosed with a sleep disorder. Information about the type or severity of the sleep disorder for each subject was not given to us. Therefore, we presumed there was a high possibility that the data we obtained from each subject statistically differed from each other.

So for a final experiment we selected data from two randomly selected subjects to use separately as test sets for our best performing method. The subject IDs selected were 19 and 15. The goal was to get an accurate evaluation of how the method would perform for different subjects.

Chapter 7

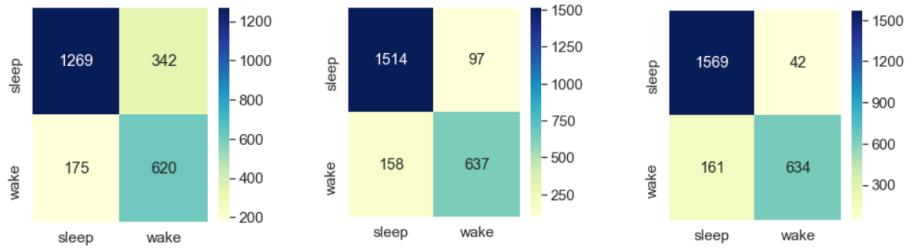
Results and Discussion

This chapter presents the final results for each of our experiments and provides discussions of what is presented.

7.1 Supervised Learning Methods

As mentioned in the previous chapter, to begin our experiments we first implemented and tested DT, RF, and XGB classifiers using our data. The purpose of this was to create a performance baseline. The resulting confusion matrices after training and testing can be seen in Figure 7.1. The performance scores can be seen in Table 7.1.

As seen from the results XGB performed the best out of the three supervised learning methods. This is why we chose to use XGB to test different feature selections. The goal was to obtain an insight into our feature selection. The resulting confusion matrices after training and testing with feature selection can be seen in Figure 7.2. The performance scores can be seen in Table 7.2.

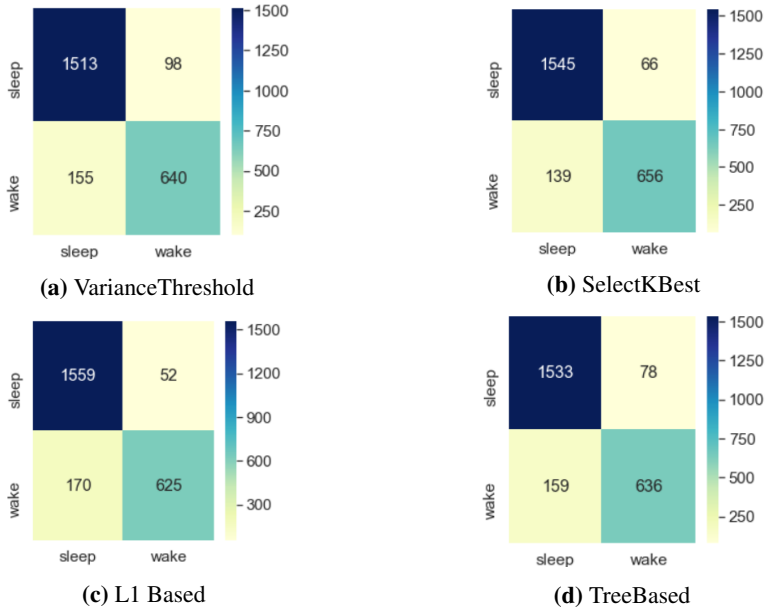


(a) DT **(b)** RF **(c)** XGB

Figure 7.1: Confusion matrices - Supervised learning methods.

Supervised Learning				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.7851	0.7877	0.7799	0.7838
Random Forest	0.8940	0.9398	**0.8013**	0.8678
XGBoost	**0.9156**	**0.9739**	0.7975	**0.8813**

Table 7.1: Performance results - Supervised learning methods.



(a) VarianceThreshold **(b)** SelectKBest

(c) L1 Based **(d)** TreeBased

Figure 7.2: Confusion matrices - XGB with feature selection

XGBoost				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
All features	**0.9156**	**0.9739**	0.7975	0.8813
Variance Threshold	0.8948	0.9392	0.8050	0.8695
SelectKBest	0.9148	0.9590	**0.8282**	**0.8896**
L1-based	0.9077	0.9677	0.7862	0.8722
Tree-based	0.9015	0.9516	0.8000	0.8725

Table 7.2: Performance results - XGB with feature selection.

7.1.1 Discussion

As shown in Table 7.1 XGB is clearly the best performing classifier of the three with the highest accuracy, sensitivity and G-Mean scores. RF performs best when it comes to specificity and has a G-Mean score close to XGB. DT has clearly the worst performance.

The confusion matrices in Figure 7.1 show that the XGB classifier only mislabelled 42 sleep epochs as wake. RF and DT classifiers mislabelled 97 and 342 sleep epochs respectively. However, XGB also mislabelled 161 wake epochs as sleeps. Both RF and DT classifiers had similar performance for wake classification with 158 and 175 mislabelled wake epochs, respectively. This does not come as a huge surprise as the data sets have been gathered from subjects diagnosed with a sleep disorder.

Subjects with sleep disorders often lie awake in bed with minimal movement for extended periods of time. These wake periods can easily be confused with sleep periods as they both contain similar movement patterns. This means it might be difficult for a sleep/wake classifier to properly distinguish between sleep and wake epochs. Because the classifiers had clear problem with this is could therefore indicate that our feature selection is not completely adequate for sleep/wake classification for subjects with sleep disorders. Additional or less features could potentially help provide better results.

The confusion matrices presented in Figure 7.2 and the results shown in Table 7.2 show how the XGB classifier performed when feature selection algorithms where used. The results show that when using all features the algorithm had the highest accuracy and sensitivity scores. Specificity and G-mean scores were the highest for when *SelectKBest* feature selection was used. Specificity was also slightly raised for when using *VarianceThreshold* and *Tree-based* feature selection, but both sensitivity and G-mean scores were lowered.

Even though using feature selection methods might have provided an slight increase in specificity, we did not find it the increase high enough to balance out the lowered accuracy and sensitivity scores. Based on this, we decided to keep all selection of a 100 features for our further experiments with binary sleep/wake classification.

7.2 Multiclass Classification

As mentioned in the previous chapter, we completed several experiments for multiclass classification. The experiments included testing with all arousal types found in our annotations with or without feature selection, testing with only PLM arousals and testing with balanced PLM arousals. The resulting confusion matrices for all multiclass experiments can be found in Appendix C. The remaining results from these experiments are presented as follows.

7.2.1 Results - All Arousal Types

The first part of our experiments with multiclass classification was the training and testing DT, RF, and XGB classifiers using data with sleep, wake and arousal labels. The arousal labels represented all arousal types provided in the arousal annotations. The performance scores can be seen in Table 7.3.

DT			
	Arousal	**Sleep**	**Wake**
Precision	0.2566	0.6733	0.5265
Recall	0.4018	0.4327	0.6418
Accuracy	0.4780		
RF			
	Arousal	**Sleep**	**Wake**
Precision	0.2749	0.7134	0.6964
Recall	0.2288	0.7460	0.7380
Accuracy	0.6247		
XGB			
	Arousal	**Sleep**	**Wake**
Precision	0.3947	0.7112	0.7277
Recall	0.0811	0.9295	0.7977
Accuracy	0.7007		

Table 7.3: Performance results for Multiclass classification - all arousal types.

7.2.2 Results - XGB with Feature Selection

The results from of our second experiment with multiclass classification where XGB was utilized along with four feature selection algorithms can be seen in Table 7.4.

7.2.3 Results - PLM Arousals

Our third experiment with multiclass classification involved only using PLM arousals in our data set. The performance scores can be seen in Table 7.5.

7.2.4 Results - PLM Arousals Balanced

The last experiment with multiclass classification involved only using PLM arousals in our data set where the arousal instances had been duplicated to achiece a more balanced data set.The performance scores can be seen in Table 7.6.

XGB -	**All features**		
	Arousal	**Sleep**	**Wake**
Precision	0.3947	0.7112	0.7277
Recall	0.0811	0.9295	0.7977
Accuracy	0.7007		

XGB -	**Variance Threshold**		
	Arousal	**Sleep**	**Wake**
Precision	0.2625	0.6912	0.6993
Recall	0.0757	0.8646	0.7944
Accuracy	0.6650		

XGB -	**SelectKBest**		
	Arousal	**Sleep**	**Wake**
Precision	0.2424	0.7011	0.7005
Recall	0.0432	0.8870	0.8458
Accuracy	0.6820		

XGB -	**L1-based**		
	Arousal	**Sleep**	**Wake**
Precision	0.3273	0.7030	0.6960
Recall	0.0649	0.8894	0.8275
Accuracy	0.6837		

XGB -	**Tree-based**		
	Arousal	**Sleep**	**Wake**
Precision	0.3214	0.6964	0.7075
Recall	0.0649	0.9022	0.7944
Accuracy	0.6820		

Table 7.4: Performance results for XGB Multiclass classification - all arousal types with feature selection.

DT			
	Arousal	**Sleep**	**Wake**
Precision	0.0521	0.8164	0.4779
Recall	0.2051	0.6103	0.5915
Accuracy	0.5914		

RF			
	Arousal	**Sleep**	**Wake**
Precision	0.0000	0.8962	0.8022
Recall	0.0000	0.9417	0.7923
Accuracy	0.8658		

XGB			
	Arousal	**Sleep**	**Wake**
Precision	0.0000	0.8924	0.8081
Recall	0.0000	0.9461	0.7883
Accuracy	0.8674		

Table 7.5: Performance results for Multiclass classification - PLM arousals.

DT			
	Arousal	**Sleep**	**Wake**
Precision	0.0205	0.8806	0.6122
Recall	0.0769	0.6792	0.7377
Accuracy	0.6775		

RF			
	Arousal	**Sleep**	**Wake**
Precision	0.0000	0.8984	0.8281
Recall	0.0000	0.9580	0.7964
Accuracy	0.8778		

XGB			
	Arousal	**Sleep**	**Wake**
Precision	0.0813	0.9055	0.8148
Recall	0.1282	0.9123	0.7514
Accuracy	0.8379		

Table 7.6: Performance results for Multiclass classification - PLM arousals with balanced data set.

7.2.5 Discussion

The performance results presented in Table 7.3 clearly shows that all three classifiers failed to accurately classify the data. The highest overall accuracy score was for XGB which had an accuracy of 0.7007, while the lowest accuracy score was for DT at 0.4780. It is quite clear that the classifiers are not able to distinguish between the classes using our feature set.

When using feature selection algorithms for the XGB classifiers the performance results for the arousal class declined even more. None of the arousal precision and recall scores for any of the feature selection algorithms beat the original scores for when all features where used. Overall, the only score that had an increase was the wake recall when using *SelectKBest*. Otherwise, none of the other scores was higher than for when using all features. One reason for this can be that our feature selection is not suited to properly distinguish between sleep, wake and arousal. More specifically, if few or none of our features are able to represent the clear differences between the three classes a smaller selection of them will not help improve the classification performance.

With regards to only using PLM arousals, the precision and recall scores for the arousal class presented in Table 7.5 show that both RF and XGB classifiers failed to label a single instance as ”arousal”. The perceived reason behind this is the low amount of PLM arousal instances. We already mentioned previously how an unbalanced data set can result in classifiers overlooking the class with the fewest instances to achieve a higher overall accuracy score. It is very likely that this is the case for both RF and XGB classifiers as even though precision and recall for arousal is at 0 the overall accuracy it at 86%. The other performance scores in Table 7.5 further underlines this possibility as both sleep and wake precision and recall scores for both RF and XGB have significantly increased compared to when all arousal types was used. The only exception is the value of wake recall for XGB where there was instead a slight decrease. Still, the results give a clear indication that the number of PLM arousal instances is not adequate for accurate classification of arousals. This was the main reason why we decided to also test using a more balanced data set.

The precision and recall scores for the arousal class presented in Table 7.5 show that after duplicating the PLM arousal instances the RF classifier still failed to label a single instance as ”arousal”. On the other hand, if you compare Tables 7.5 and 7.6 XGB’s performance scores for the arousal class increased after the duplication. The precision and recall scores increased from 0.0 to 0.0813 and 0.1282, respectively. With regards to DT, the precision and recall scores for both arousal and wake instead actually decreased. The arousal scores went from 0.0521 and 0.2051 to 0.0205 and 0.769, respectively. These results indicate that a more balanced results could possibly help improve arousal classification but because the results varied from classifier to classifier we cannot say this with certainty.

It is also possible that the varied and low results are caused by inadequate representation of arousals in our feature set. The precision and recall scores in general for the arousal

class is quite low in all of our experiments. The highest value either of the scores ever get is about 0.40. In addition, based on the confusion matrices shown arousal instances are more often labelled as either sleep or wake than their actual class. This clearly suggests that our feature selection is not suited to be used to differentiate between arousal and sleep or wake.

Something that should be noted is that it is possible that the PLM movements of the feet was to small to be accurately picked up by our sensors. It is also possible that if a PLM movement occurred for the opposite foot of where our thigh sensor was placed the sensor might not have picked up the movement at all. This could also be correct for other arousal types, such as Leg Movement (LM) arousals. This could mean that our sensor placements could be, in addition to our feature selection, not optimal for our arousal classification.

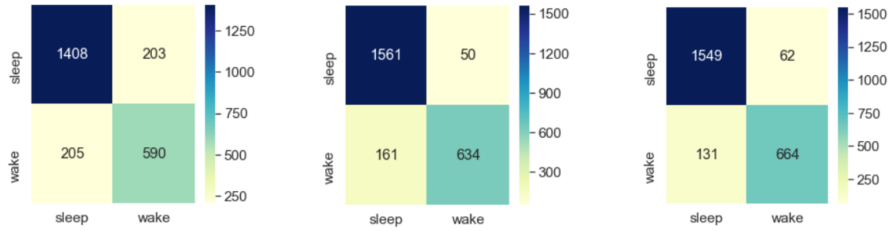
In addition, as mentioned, all arousal annotations for all arousal types were made automatically and cannot be completely relied on. This means that it is not possible at this time to accurately determine the suitability of our data collection set up and feature selection. It is possible that our feature selection represent arousals better than what has been shown in the results, but error and mistakes in the annotations prohibited this from being shown. Therefore, further testing with proper arousal data and potentially new features is needed to determine if sleep/wake/arousal multiclass classification can help improve the performance of sleep pattern detection.

7.3 Co-training with Single-view

For co-training with single-view (CoSV) we conducted two experiments. In the first one we used data from 5 subjects as the initial labelled data. In the second experiment we used data from all 17 subjects in our training set at the initial labelled data. The results for each experiment can be found below.

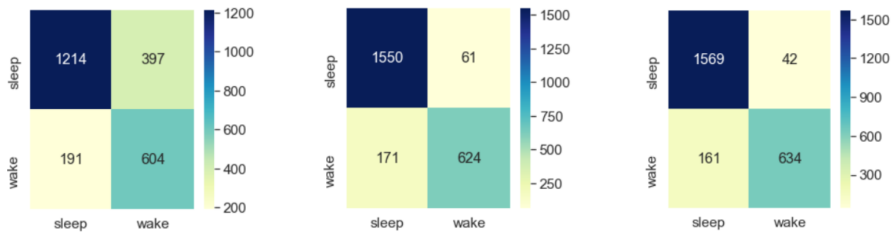
7.3.1 Results

The confusion matrices shown in Figure 7.3 show the result of training the CoSC method using 5 subjects as initial labelled data. The matrices shown in Figure 7.4 show the results after using all training set subjects as initial labelled data. The performance scores from both experiments are shown in Table 7.7.



(a) DT **(b)** RF **(c)** XGB

Figure 7.3: Confusion matrices - CoSV with 5 subjects as labelled data.



(a) DT **(b)** RF **(c)** XGB

Figure 7.4: Confusion matrices - CoSV with all subjects as labelled data.

CoSV	**- 5 subjects**			
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.8304	0.8740	0.7421	0.8054
Random Forest	0.9123	**0.9690**	0.7975	0.8791
XGBoost	**0.9198**	0.9615	**0.8352**	**0.8961**
CoSV	**- all subjects**			
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.7556	0.7536	0.7597	0.7567
Random Forest	0.9036	0.9621	0.7849	0.8690
XGBoost	**0.9156**	**0.9739**	**0.7975**	**0.8813**

Table 7.7: Performance results - Co-training with Single-view

7.3.2 Discussion

When comparing the results for CoSV shown in Table 7.7 with the original baseline supervised learning results in Table 7.1 several of the performance scores of CoSV where close too, equal too or higher than the baseline scores. CoSV with 5 subjects as initial labelled data (CoSV-5) performed in general better than when using all training set subjects (CoSV-all). DT, RF and XGB all had higher overall scores. The best results was for when using CoSV-5 with XGB. When compared to the best results from the baseline results the accuracy went from 0.9156 to 0.9198, specificity from 0.7975 to 0.8352 and G-mean from 0.8813 to 0.8961. Only sensitivity had a slight decrease from 0.9739 to 0.9615.

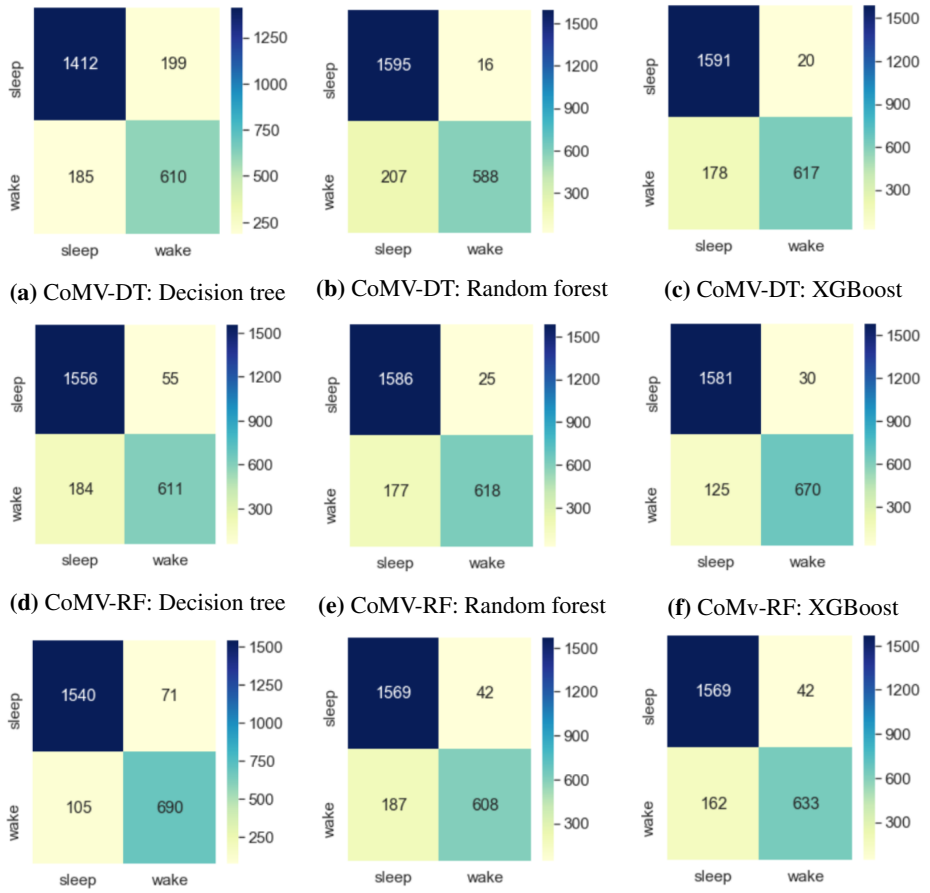
The fact that both CoSV-5 and CoSV-all had such positive results give a clear indication that using semi-supervised ensemble methods can clearly improve the performance scores for sleep pattern detection on the HUNT4 data. As shown, using only data from 5 subject as the only labelled data together with large amount of unlabelled data gave the best results. Using this semi-supervised method can also therefore decrease the necessary amount of labelled data needed to train an adequate classifier.

However, the fact that when starting with less subjects (CoSV-5) results in a better performance than for using all subjects (CoSV-all) it gives a strong indication that there exists large variations in the available labelled sleep data and overfitting might have occurred. This is not completely surprising as people with sleep disorders can have significantly different sleep patterns, especially if the severity of the disorder differs. Since CoSV-5 performed best it might indicate that the five randomly selected subject might share a more similar sleep pattern with the test set subjects than the average among all subjects. This variation in the data set could mean that the method might not generalize well and might perform different for new unseen instances.

It is also possible that our PL data set is too small for any classifier to generalize well, as it only consists of data collected from 19 subjects. Section 3.3.1 provides an overview of the number of subjects used for data collection in our introduced collection of related work using machine learning methods. As seen, the number of subjects used in the papers varies. The lowest number of subjects used being 20 and the highest number 354. Since our number of subjects is below this range it might be necessary to obtain data from more subjects for our experiments in order to obtain a more accurate evaluation of our methods.

7.4 Co-training with Multi-view

For co-training with multi-view (CoMV) we also conducted two experiments using the same initial selection for labelled data as with CoSV. We ran the CoMV method using DT, RF, and XGB classifiers separately and trained and tested the resulting data sets using the same classifiers. The results for each experiment can be found below.



(a) CoMV-DT: Decision tree **(b)** CoMV-DT: Random forest **(c)** CoMV-DT: XGBoost

(d) CoMV-RF: Decision tree **(e)** CoMV-RF: Random forest **(f)** CoMv-RF: XGBoost

(g) CoMV-XGB: Decision tree **(h)** CoMV-XGB: Random forest **(i)** CoMV-XGB: XGBoost

Figure 7.5: Confusion matrices - CoMV with five subjects as labelled data.

7.4.1 Results - 5 training subjects

The confusion matrices after training and testing with the labelled data sets received from using the CoMV method (with 5 subjects as initial labelled data) can be seen in Figure 7.5. The performance results are shown in Table 7.8.

CoMV-DT				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.8404	0.8765	0.7673	0.8201
Random Forest	0.9073	**0.9901**	0.7396	0.8557
XGBoost	**0.9177**	0.9876	**0.7761**	**0.8755**
CoMV-RF				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.9007	0.9659	0.7686	0.8618
Random Forest	0.9160	**0.9845**	0.7774	0.8748
XGBoost	**0.9356**	0.9814	**0.8428**	**0.9094**
CoMV-XGB				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	**0.9268**	0.9559	**0.8679**	**0.9109**
Random Forest	0.9048	**0.9739**	0.7648	0.8630
XGBoost	0.9152	**0.9739**	0.7962	0.8806

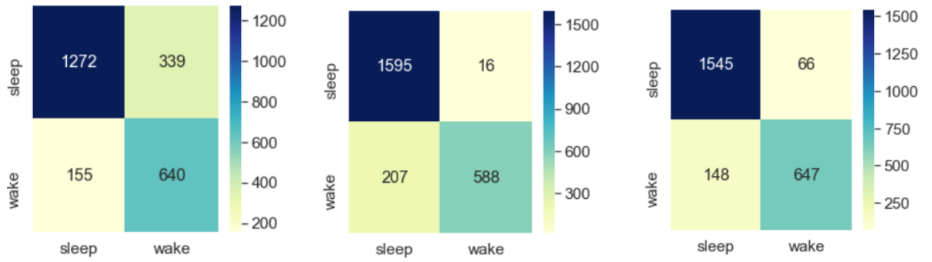
Table 7.8: Performance results - Co-training with Multi-view with 5 subjects as labelled data

7.4.2 Results - All 17 subjects

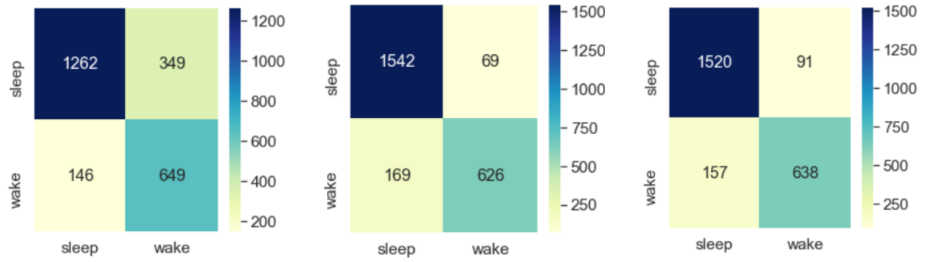
The confusion matrices after training and testing with the labelled data sets received from using the CoMV method (with all 17 subjects as initial labelled data) can be seen in Figure 7.6. The performance results are shown in Table 7.9.

CoMV-DT				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.7947	0.7896	0.8050	0.7973
Random Forest	0.8994	0.9497	0.7975	0.8703
XGBoost	**0.9111**	**0.9590**	**0.8138**	**0.8835**
CoMV-RF				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.7943	0.7834	**0.8164**	0.7997
Random Forest	**0.9011**	**0.9572**	0.7874	0.8682
XGBoost	0.8969	0.9435	0.8025	**0.8702**
CoMV-XGB				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.8229	0.8113	0.8465	0.8287
Random Forest	0.9123	**0.9503**	0.8352	0.8909
XGBoost	**0.9140**	0.9354	**0.8704**	**0.9024**

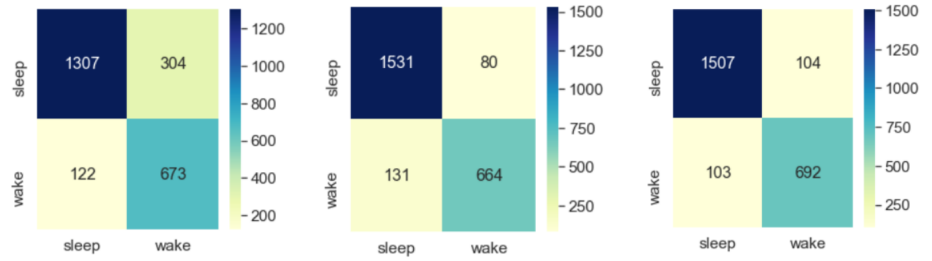
Table 7.9: Performance results - Co-training with Multi-view with all 17 subjects as labelled data



(a) CoMV-DT: Decision tree **(b)** CoMV-DT: Random forest **(c)** CoMV-DT: XGBoost



(d) CoMV-RF: Decision tree **(e)** CoMV-RF: Random forest **(f)** CoMV-RF: XGBoost



(g) CoMV-XGB: Decision tree **(h)** CoMV-XGB: Random forest **(i)** CoMV-XGB: XGBoost

Figure 7.6: Confusion matrices - CoMV with all subjects as labelled data.

7.4.3 Discussion

Table 7.8 shows that CoMV outperforms CoSV with a highest accuracy score of 0.9356. This score came from once again using only 5 subjects as initial labelled data (CoMV-5). More specifically, it was the accuracy score from using data created by the CoMV-RF method (CoMV using only RF classifiers) to train and test an XGB classifier. The highest specificity and g-mean scores for CoMV-5 came from using data created by the CoMV-XGB method to train and test a decision tree classifier. The scores were 0.8679 and 9109, respectively. The accuracy score was also the second highest for CoMV at 0.9268.

The performance scores for CoMV using all 17 training set subjects as initial labelled data (CoMV-all) are shown in Table 7.9 and shows that it did not have as good performance results as CoMV-5. The highest accuracy, sensitivity, specificity and g-mean scores were at 0.9140, 0.9590, 0.8704 and 0.9024, respectively. When compared to the baseline supervised learning results shown in Table 7.1 CoMV-all only outperformed the results with regards to specificity and g-mean. However, the specificity score is the highest so far for our experiments. Despite these varied results, the positive results from using CoMV-5 clearly shows that semi-supervised ensemble methods can be a great asset to improving sleep/wake classification.

However, the fact that using only 5 subjects as initial labelled data outperforms using all 17 once again indicates that there exists an significant variation in the data collected from each subject in the PL data set. This also further gives proof to the suggestion that the methods might generalize better for unseen instances if more data from other subjects was added to the training set.

7.5 Supervised Multi-view

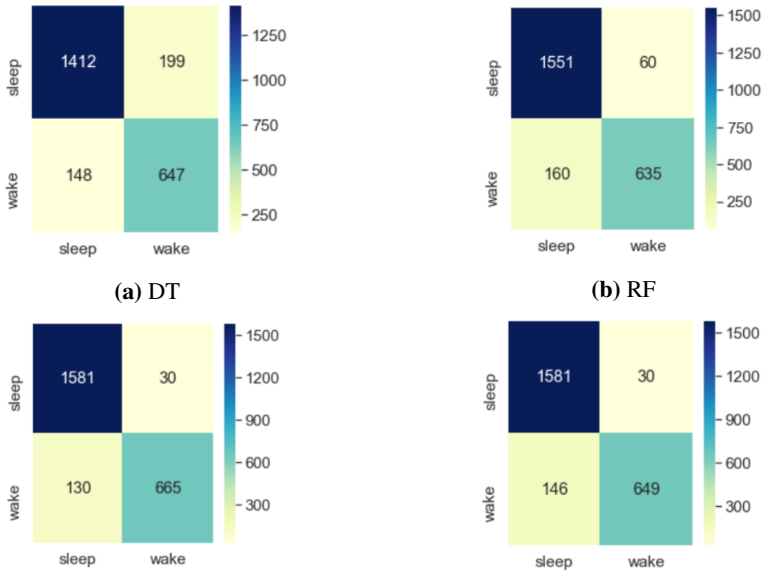
For our supervised multi-view (SuMV) we ran the method first using DT, RF, and XGB classifiers separately and then all together (as previously described in section 6.7.1). The results of the experiments with SuMV can be found below.

7.5.1 Results

The resulting confusion matrices after training and testing the SuMV method using different classifiers can be seen in Figure 7.7. The specific performance results are shown in Table 7.10.

7.5.2 Discussion

The results presented show that the SuMV method performed very well with the exception of when the DT classifiers where used. If we disregard the results for DT shown in the



(a) DT **(b)** RF

(c) XGB **(d)** All

Figure 7.7: Confusion matrices - Supervised Multi-view

SuMV	Accuracy	Sensitivity	Specificity	G-Mean
Decision Tree	0.8558	0.8765	0.8138	0.8446
Random Forest	0.9086	0.9628	0.7987	0.8769
XGBoost	**0.9335**	**0.9814**	**0.8365**	**0.9060**
All	0.9268	**0.9814**	0.8164	0.8951

Table 7.10: Performance results - Supervised Multi-view method

confusion matrices in Figure 7.7, then SuMV only mislabelled 45±15 sleep epochs as wake and 145±15 wake epochs as sleep. The low number of mislabelled sleep epoch is quite promising and these results are reflected in the high sensitivity scores presented in Table 7.10, with the best scores at 0.9814. The highest scores overall are for when XGB classifiers are used with scores at 0.9335, 0.9814, 0.8365, and 0.9060 for accuracy, sensitivity, specificity and g-mean, respectively.

However, all the best performance results of SuMV only come close to the best performance scores for CoMV (see Table 7.8). This could be another indication that getting a larger data set could help the improve the performance of our methods. Despite that possibility, based on the labelled data we do have, the results from the SuMV method is very encouraging. They do provide strong proof that using an ensemble of supervised learning methods can be used successfully to improve the overall performance results for

sleep pattern detection. Nevertheless, we still wanted to examine if the results could be improved even further.

7.6 Supervised Multi-view with Clustering

For supervised multi-view (SuMV) with clustering we experimented using two different clustering methods: K-means clustering and agglomerative hiearchical clustering. Each method was testing first using the training and test set as specified in section 6.2. Secondly they were also trained and tested using the data sets resulting from using the CoSV and CoMV methods. We chose the data sets where 5 subjects were set as initial labelled data had been used as they had the best performance. The results from the experiments is shown in the following subsections.

7.6.1 Results - SuMV with K-Means Clustering

All resulting confusion matrices after training and testing SuMV with K-means clustering can be found in Appendix C. The specific performance results after training and testing SuMV with K-means clustering using our standard training and test sets can be seen in Figure are shown in Table 7.11. The performance results after training and testing SuMV with K-means clustering using our CoSV and CoMV data sets are shown in Table 7.12.

SuMV	**with K-Means**			
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.9256	0.9863	0.8025	0.8897
Random Forest	0.9339	**0.9882**	0.8239	0.9023
XGBoost	0.9310	0.9851	0.8214	0.8995
All	**0.9372**	0.9876	**0.8352**	**0.9082**

Table 7.11: Performance results - Supervised Multi-view with K-means clustering

7.6.2 Results - SuMV with Agglomerative Hiearchical Clustering

All resulting confusion matrices after training and testing SuMV with agglomerative hiearchical clustering (AHC) can be found in Appendix C. The specific performance results after training and testing SuMV with AHC using our standard training and test sets are shown in Table 7.13. The performance results after training and testing SuMV with AHC using our CoSV and CoMV data sets are shown in Table 7.14.

CoSV				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.9202	0.9932	0.7723	0.8758
Random Forest	0.9235	0.9926	0.7836	0.8819
XGBoost	**0.9352**	0.9907	**0.8226**	**0.9028**
All	0.9202	**0.9950**	0.7686	0.8745
CoMV-DT				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	**0.9160**	0.9932	**0.7597**	**0.8687**
Random Forest	0.8990	0.9963	0.7019	0.8362
XGBoost	0.9106	0.9957	0.7384	0.8574
All	0.9023	**0.9975**	0.7094	0.8412
CoMV-RF				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.9273	0.9913	0.7975	0.8891
Random Forest	0.8944	0.9448	0.7925	0.8653
XGBoost	**0.9397**	**0.9957**	**0.8264**	**0.9071**
All	0.9281	0.9932	0.7962	0.8893
CoMV-XGB				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.9156	0.9913	0.7623	0.8693
Random Forest	0.9098	**0.9969**	0.7333	0.8550
XGBoost	**0.9451**	0.9913	**0.8516**	**0.9188**
All	0.9339	0.9950	0.8100	0.8978

Table 7.12: Performance results - Supervised Multi-view with K-means clustering using data sets from semi-supervised methods.

SuMV with	**AHC**			
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.9090	0.9832	0.7585	0.8636
Random Forest	0.9256	**0.9894**	0.7962	0.8876
XGBoost	0.9335	0.9870	0.8252	0.9024
All	**0.9339**	0.9826	**0.8352**	**0.9059**

Table 7.13: Performance results - Supervised Multi-view with agglomerative hiearchical clustering

CoSV				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.9210	0.9894	0.7824	0.8798
Random Forest	**0.9347**	0.9944	**0.8138**	**0.8996**
XGBoost	0.9302	0.9919	0.8050	0.8936
All	0.9177	**0.9957**	0.7597	0.8697
CoMV-DT				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	**0.9323**	0.9851	**0.8282**	**0.9016**
Random Forest	0.9202	0.9944	0.7698	0.8749
XGBoost	0.9106	**0.9975**	0.7346	0.8560
All	0.9181	0.9957	0.7610	0.8705
CoMV-RF				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.9210	0.9727	0.8164	0.8911
Random Forest	0.9002	0.9640	0.7711	0.8622
XGBoost	**0.9431**	**0.9957**	**0.8365**	**0.9126**
All	0.9289	0.9932	0.7987	0.8907
CoMV-XGB				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.8944	0.9913	0.6981	0.8319
Random Forest	0.9397	**0.9963**	0.8252	0.9067
XGBoost	**0.9468**	0.9913	**0.8566**	**0.9215**
All	0.9298	0.9932	0.8013	0.8921

Table 7.14: Performance results - Supervised Multi-view with AHC using data sets from semi-supervised methods.

7.6.3 Discussion

When comparing the results found in Table 7.11 with the results in Table 7.12 it shows that when performing SuMV with K-means clustering using data sets obtained from our proposed semi-supervised methods as training data the performance scores where overall higher than for when using our standard training set. This also held true for the SuMV with AHC method (see Table 7.13 and Table 7.14).

The best results for SuMV with k-means clustering was when the classifier was XGBoost and the CoMV-XGB data set was used as training data. The accuracy was at 0.9451, sensitivity at 0.9913, specificity at 0.8516, and g-mean at 0.9188. The best results for SuMV with AHC was also when the classifier was XGBoost and the CoMV-XGB data set was used as training data. The accuracy was at 0.9468, sensitivity at 0.9913, specificity at 0.8566, and g-mean at 0.9215. These results once again shows the potential of using ensemble methods for improving sleep/wake classification performance results.

If you also compare the results of SuMV found in Table 7.10 with the results of SuMV with clustering, Tables 7.11 and 7.13, you can clearly see that for all methods using clustering provides a higher overall accuracy for all methods. When using SuMV with DT the was only as 0.8558. For both SuMV with K-means and SuMV with AHC this value increased to 0.9256 and 0.9090, respectively. These results are strong indications that using SuMV with clustering results in a more stable classifier for sleep/wake classification.

In all the results presented so far the results from CoMV-all showed the highest score for specificity. Consequently, using the CoMV-all data sets might have been a better idea to use for these experiments with clustering. It is possible the results might have given us a higher overall specificity and G-mean score and provided a more even ability to classify both classes. As it stands, the sensitivity scores for our current results are quite high, while specificity scores still remains below 0.88.

7.7 Comparison of Methods

To get a better overview of how our methods performed we decided to compare the best results from each method. We excluded the results from our experiments with multiclass classification as the scores were quite low and could not compete with the performance of the other methods. It also enabled easier comparison to only focus on the results for binary sleep/wake classification. Consequently, Table 7.15 presents the baseline results from the supervised learning methods along with the best results from each of our experiments.

7.7.1 Discussion

As seen in Table 7.15 the best performing method was our the Supervised Multi-view with Agglomerative Hierarchical Clustering method using XGBoost classifiers and the CoMV-

Baseline - Supervised learning methods				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
Decision Tree	0.7851	0.7877	0.7799	0.7838
Random Forest	0.8940	0.9398	**0.8013**	0.8678
XGBoost	**0.9156**	**0.9739**	0.7975	**0.8813**
Proposed Methods				
	Accuracy	**Sensitivity**	**Specificity**	**G-Mean**
CoSV-5: XGBoost	0.9198	0.9615	0.8352	0.8961
CoMV-RF-5: XGBoost	0.9356	0.9814	0.8528	0.9094
CoMV-XGB-5: Decision tree	0.9268	0.9559	**0.8679**	0.9109
SuMV: XGBoost	0.9335	0.9814	0.8365	0.9060
SuMV-kMeans: XGBoost (CoMV-XGB)	0.9451	**0.9913**	0.8516	0.9188
SuMV-AHC: XGBoost (CoMV-XGB)	**0.9468**	**0.9913**	0.8566	**0.9215**

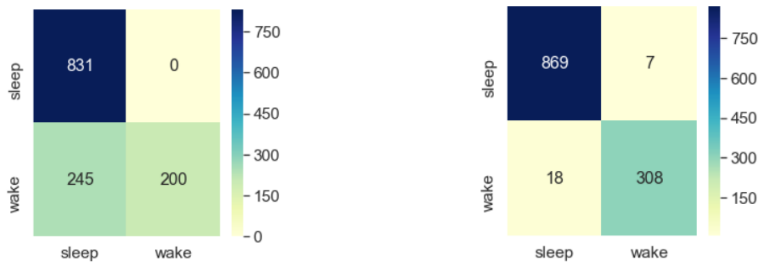
Table 7.15: Performance results - Comparing proposed methods for binary sleep/wake classification.

XGB data set as training data. The accuracy, sensitivity, specificity and g-mean scores were at 0.9468, 0.9913, 0.8566, and 0.9215, respectively. One thing that also stands out in Table 7.15 is that for each of our best performing proposed methods the XGB classifier was used in some way. Therefore, it is easy to say that between DT, RF, and XGB the XGB classifier is clearly the most suitable classifier for our purposes.

The performance results of our proposed methods clearly shows the usability of ensemble methods for sleep/wake classification. When compared to the baseline results using ensemble methods, at best, increased the accuracy score by 3.12%, sensitivity by 1.74%, specificity by 6.66%, and g-mean by 4.02%. This is a notable increase in performance and clearly shows that our methods are a viable option for sleep/wake classification.

Something that we mentioned previously is that we were not given any information on what type of sleep disorder our subjects from the PL data set had or what the severity was in each case. If the subject had different types and/or severities is might have resulted in large variations in the data set. And, as it stands, we only obtained data from 19 subjects, which might not have been enough for our classifiers to generalize well. Consequently, if the training data set is too little there can be a big variance in performance when confronted with new unseen data. The fact that SuMV with clustering performed better when using a larger data set as training data gives a good indication that this might be the case. Furthermore, it also increases the possibility that overfitting has occurred.

As we mentioned, when testing the SuMV with clustering methods using data sets from CoSV and CoMV methods as training data we decided to use the data sets were only five subjects were set as the initial labelled data. This means that the resulting data sets reflect the statistics and variance found in the data collected from these five subjects. It is



(a) Subject ID 15 **(b)** Subject ID 19

Figure 7.8: Confusion matrices - Final Experiment

possible that we randomly selected five subjected that had a higher average similarity to our test set subjects than what entire training set does. If this is the case, the performance of our proposed methods might decrease significantly when classifying data from a new subject. However, to accurately determine if overfitting has occurred further testing using new subjects need to be conducted.

7.8 Final Experiment

As mentioned previously, for our final experiment we selected data from two randomly selected subjects in our training set to use separately as test sets for our best performing method. The subject IDs selected were 15 and 19, and the goal was to obtain an accurate evaluation of how our method would perform for different subjects.

As seen in Table 7.15, our best performing method was the SuMV with Agglomerative Hierarchical Clustering method using XGBoost classifiers and the CoMV-XGB data set as training data. Because the CoMV data set was partially based on the unlabelled data from our selected test subjects we decided not to use the CoMV data set in this experiment.

Instead, data from our new test subjects, 15 and 19, would be removed from the training data and set as the test sets. The original test set (14 and 18) were added to the training data so we once again had a training set consisting of data from 17 subjects. The training and test sets were then used to test the data from each test set subject separately using the SuMV-AHC method with XGBoost classifiers.

7.8.1 Results

The resulting confusion matrices from testing with subjects 15 and 19 is in Figure 7.8. The performance results are presented in Table 7.16.

Final test				
Subject ID	**Accuracy**	**Sensitivity**	**Specificity**	**G-Mean**
15	0.8080	1.000	0.4494	0.6704
19	0.9792	0.9920	0.9448	0.9681

Table 7.16: Performance results - Final experiment

7.8.2 Discussion

As seen from both Figure 7.8 and Table 7.16 the performance results for each subject varies drastically. Testing using data from subject 15 only results in an accuracy score of 0.8080, while using data from subject 19 results in an accuracy score of 0.9792. This is a significant difference in values. It must be mentioned that since we switched data from between the test and train sets it is possible that the original split performs slightly different for completely new instances. However, the results we do have is a clear indication that our method fails to generalize well for new data.

From looking at the overview of the available data for each subject seen in Table 5.4 (section 5.1.2) it can be seen that the data from subject 15 has a higher amount of wake instances than the data from subject 19. This suggests that subject 15 struggled more with falling asleep than subject 19. The confusion matrix for subject 15 (Figure 7.8a) supports this suggestion as only wake instances was mislabelled and no sleep instances. This gives an indication that our methods are more suitable and will generalize better for data from perceived healthier subjects. To completely test this theory it will be necessary to test using data from a healthy subject. But as we had not obtained any such data during this research it was not possible for us to conduct this test.

It is also possible that the variation in performance is the result of having a too small training set. As we have already mentioned, based on related work, it is more common to use a larger data set when training a sleep/wake classifier. Our data set only contains data gathered from 19 individuals with one night of data provided for each. Data from 17 of them was used to train our classifiers. With regards to the related work presented in Chapter 3 focusing on machine learning methods (sections 3.2.2 and 3.2.3) the lowest number of subjects used in one of the presented papers was 20, which is almost equal to our number. However, in section 3.3.1 it is also shown that 2 nights of data was provided for each of the 20 subjects. This clearly suggests that our data set is too small for our purpose and that a larger data set is most likely required for our methods to generalize well.

Chapter 8

Conclusion and Future Work

In this chapter we present our conclusion to our research along with our contributions and suggest some hypothetical areas of future work that can be done based on our research.

8.1 Conclusion

As a result of our research we have developed several methods for sleep pattern detection using machine learning that can be used in the HUNT4 study. The main methods consist of an ensemble method for binary sleep/wake classification that can distinguish between sleep and wake epochs using features based on raw accelerometer data collected from two accelerometer sensors placed on a subject's mid lower back and thigh. In addition, we also proposed methods for sleep pattern detection using multiclass classification. However, the multiclass models failed to accurately distinguish between the three classes (sleep, wake, and arousal). The highest precision score for the arousal class only reached a value of 0.3947. Our proposed ensemble methods for binary sleep/wake classification performed better with sensitivity and specificity scores above 0.9700 and 0.8000, respectively.

All our proposed ensemble methods for binary sleep/wake classification outperformed basic DT, RF and XGB classifiers. Only the CoSV method failed to achieve a higher sensitivity score than the XGB classifier, while all of the other proposed methods achieved improved values for all performance scores. Our best performing method was the Supervised Multi-view with Agglomerative Hierarchical Clustering method using XGB classifiers and training data obtained through the CoMV semi-supervised method. The accuracy, sensitivity, specificity, and g-mean scores were at 0.9451, 0.9913, 0.8516, and 0.9188, respectively. This method combined supervised, unsupervised, and semi-supervised learning to obtain its results. The results showcases the strength of ensemble methods and how they can be of good use for the HUNT4 study.

Despite the promising results, there were also indications that overfitting had occurred for our models and the results from our final experiment showed that the best performing method failed to generalize well. Despite this, we would say that our proposed ensemble methods for sleep pattern detection using machine learning are quite promising and can be of use in the field of sleep pattern detection and are especially applicable to the HUNT4 study.

8.2 Contributions

Based on our research goals and questions, which are stated in Chapter 1, our research have resulted in two main contributions. The first and most important contribution is our proposed methods for sleep pattern detection which can distinguish between sleep and wake epochs with accuracy scores up to 94.68%. All methods are open sourced[1] to enable a greater understanding of our research while also allowing others to benefit from the use of our methods.

The second main contribution, based on our first research goal, it the overview of the current state-of-the-art in the field of sleep pattern detection using machine learning methods. Based on our second research goal, our research has also contributed with a insight into how both multiclass classification and ensemble methods can have an affect on sleep pattern detection methods. The ensemble methods included the use of semi-supervised learning, multiview learning and unsupervised learning, separately and combined.

8.3 Future Work

This section presents some possible research directions that can be the future next steps for improving the classification of sleep patterns on sensor data from two body-worn sensors.

8.3.1 Improved Arousal Annotations

One of the reason we could not get conclusive results for our multiclass experiments was because the arousal annotations we worked with was made automatically and not manually. Therefore we had no solid ground truth and our results could not be completely relied on. Because of the late arrival of the data used in our multiclass experiments, we could not dedicate the necessary amount of time needed to properly evaluate our set-up and features.

Consequently, one potential research direction can be to redo our experiments for multiclass (arousal) classification using more reliable arousal annotations. This will enable the researcher to properly evaluate which arousal types are more suitable for classification using our set-up and also evaluate our feature selection properly. It can also be possible to

[1] https://github.com/ailhay/SleepDetection

find new features that might be more suitable for detecting arousals, which can result in improved overall performance scores for sleep pattern detection.

8.3.2 Larger Training set

Several of the results from our experiments with binary sleep/wake classification suggests that the used training set was too small for our methods to generalize well for new instances. The results from our final experiment supported this theory along with the overview of amount of data used in related work (see section 3.3.1). Therefore, obtaining a larger labelled data set and using it to train and test our classification models can be a wise idea. A larger data set for training can potentially result in a more stable classifier that will generalize better for new instances. In addition, using a larger training set can give a better evaluation of the suitability of our models for sleep/wake classification, especially with regards to using data from subjects with diagnosed sleep disorders.

8.3.3 Personalized Classifiers

As shown in section 7.8 our best performing method failed to give similar performance results when testing data from two different test subjects. This suggested that the sleep patterns of the subjects differ from each other. Sleep disorders can manifest differently for each diagnosed patient depending on type and severity. As a consequence, it can be challenging for a classifier to accurately classify instances of data from different subjects with sleep disorders because the sleep patterns of the subjects can be very different. Which is likely what occurred for our classification models. One idea can, therefore, be to adapt classification models to specific subjects and/or groups of subjects with similar sleep patterns and/or sleep disorders.

However, a problem with this suggestion is that it might prove difficult to collect a large enough data set that reflects each subject's/group's sleep patterns. One solution to this issue can be to utilize our proposed semi-supervised methods. Getting a small amount of data that represents the subjects should be possible along with a large set of unlabelled data. Using these data sets along with a semi-supervised method can then give the necessary amount of labelled data needed to create a stable personalized classifier.

As already seen, using a smaller initial labelled data set for our semi-supervised methods resulted in better performance results (see section 7.3 and 7.4), when compared to using the entire training set as labelled data. One of our suggested reasons for this was that the smaller selection shared a larger similarity with the test subjects than the entire training set did. This supports the notion that it is possible to create personalized classifiers for subjects/groups even when only a small initial labelled data set is available.

8.3.4 Testing on Healthy Subjects

One of the problems with using data collected from subjects with sleeps disorder it that the subjects often have trouble falling asleep. This results in long periods of time where the subjects are displaying minimal movement and, for all appearances, appear to be asleep when they are in fact awake. It can therefore be difficult for a classifier to distinguish between these wake periods and actual periods of sleep, which is something that is reflected in our results. In addition, as already mentioned, when using data from subjects with sleep disorders there is a high possibility there exists significant variance in the data between subjects as sleep disorders can manifest differently for different subjects. This can be especially true when the sleep disorder type and severity varies.

On the other hand, when using data collected from healthy subjects there should not exist such a potential high variance in the data. In addition, data from healthy subjects will most likely not contain long wake periods with minimal movement. It is therefore possible that our proposed methods might prove to be more suitable for sleep pattern detection on healthy subjects. Testing this theory would require obtaining a necessary amount of labelled data from healthy subjects. However, as we did have some promising results in our research it stands to reason our methods should also work for healthy subjects. Using data from healthy subjects might even result in higher performance scores.

8.3.5 Adding Classes for Different Sleep Stages

As mentioned in section 5.1.2 instances in our PL data set was originally labelled as either wake or one of four sleep stages: N1, N2, N3, REM. 'Movement' was also an additional rare label found in the annotations which we assumed referred to movement during sleep. One potential future research direction can be to train and test our classification models using all of these annotated labels as classes. Once again, this would change the problem from a binary classification problem into a multiclass classification problem. This can require a re-evaluation of our feature selection as additional features might be needed to help distinguish between the sleep stages. However, being able to distinguish between more than just sleep and wake epochs will result in obtaining more insightful information about a subject's sleep pattern, which is what makes this research direction so interesting.

8.3.6 Adding Non-Movement Based Features

As we have already mentioned, classifiers can have problems determining sleep onset for people with sleep disorders because they often have long idle wake periods. This is especially true for research such as ours where classifiers are trained and tested utilizing only features based on movement data. Using additional features that is separate from movement can therefore help provide more accurate performance results.

One value that is detected and stored by our sensors, but not used, was skin temperature. Skin temperature may vary depending on physical exercise (Neves et al. (2015)). However, it can be viewed as non-movement based as exercise is only one of many factors that affects/determines skin temperature. Features based of skin temperature can therefore be viewed as potential non-movement based features that can be added to our feature set.

Studies have shown that the circadian rhythm of core body temperature along with skin temperature is temporally related to sleep initiation and termination (Raymann et al. (2007)). Kräuchi et al. (1999) and Kräuchi et al. (2000) have also shown that under strictly controlled settings the best physiological predictor for fast sleep onset (the transition from wakefulness into sleep) was the degree of heat loss found in the skin of the hands and feet. Adding features based on skin temperature can therefore prove to be a good idea for sleep/wake classification.

Bibliography

Ancoli-Israel, S., Cole, R., Alessi, C., Chambers, M., Moorcroft, W., Pollak, C. P., 2003. The role of actigraphy in the study of sleep and circadian rhythms. american academy of sleep medicine review paper. SLEEP 26 (3), 342–392.

Aridas, C., Kotsiantis, S., 2015. Combining random forest and support vector machines for semi-supervised learning. 19th Panhellenic Conference on Informatics, 123–128. Athens, Greece, October 01 - 03, 2015. Publication rights licensed to ACM.

Biswal, S., Kulas, J., Sun, H., Goparaj, B., Westover, M. B., Bianchi, M. T., Sun, J., 2017. Sleepnet: Automated sleep staging system via deep learning. arXiv:1707.08262 [cs.LG].

Blum, A., Mitchell, T., 1998. Combining labeled and unlabeled data with co-training. Proceedings of the eleventh annual conference on Computational learning theory, 90–100. ACM.

Borazio, M., Berlin, E., Kucukyildiz, N., Scholl, P., Laerhoven, K. V., 2014. Towards benchmarked sleep detection with inertial wrist-worn sensing unit. IEEE International Conference on Healthcare Informatics, 125–134.

Bourne, R. S., Minelli, C., Mills, G. H., Kandler, R., 2007. Clinical review: Sleep measurement in critical care patients: research and clinical implications. Critical Care, 11:226 (doi:10.1186/cc5966).

Chen, Q., Sun, S., 2009. Hierarchical multi-view fisher discriminant analysis. In: Leung C.S., Lee M., Chan J.H. (eds) Neural Information Processing. ICONIP 2009.Lecture Notes in Computer Science, vol 5864. Springer, Berlin, Heidelberg.

Cole, R. J., Kripke, D. F., Gruen, W., Mullaney, D. J., Gillin, J. C., 1992. Automatic sleep/wake identification from wrist activity. Sleep 15 (5), 461–469, https://doi.org/10.1093/sleep/15.5.461.

Duncan, S., Stewart, T., Mackay, L., Neville, J., Narayanan, A., Walker, C., Berry, S., Morton, S., 2018. Wear-time compliance with a dual-accelerometer system for capturing 24-h behavioural proles in children and adults. International Journal of Environmental Research and Public Health 15 (7), 1296; https://doi.org/10.3390/ijerph15071296.

EL-Manzalawy, Y., Buxton, O., Honavar, V., 2017. Sleep/wake state prediction and sleep parameter estimation using unsupervised classification via clustering. IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 718–723.

Enomoto, M., Endo, T., Suenaga, K., Miura, N., Nakano, Y., Kohtoh, S., Taguchi, Y., Aritake, S., Hihuchi, S., Matsuura, M., Takahashi, K., Mishima, K., 2009. Newly developed waist actigraphy and its sleepwake scoring algorithm. Sleep and Biological Rhythms 7, 17–22.

Fonseca, P., Long, X., Foussier, J., Aarts, R. M., 2013. On the impact of arousals on the performance of sleep and wake classification using actigraphy. 35th Annual International Conference of the IEEE EMBS Osaka, Japan3-7 July.

Granovsky, L., andNancy Yacovzada, G. S., Frank, Y., Fine, S., 2018. Actigraphy-based sleep/wake pattern detection using convolutional neural networks. arXiv preprint, arXiv:1802.07945 [cs.LG].

Halász, P., Terzano, M., Parrino, L., Bódizs, R., 2014. The nature of arousal in sleep. Journal of Sleep Research 13, 1–23.

Hay, A., 2018. Data analytics and healthcare: Recognition of sleep patterns on sensor data streams. Norwegian University of Science and Technology. Department of Computer Science. TDT4501 Computer Science, Specialization Project. Submitted December 2018.

Hessen, H.-O., Tessem, A. J., 2015. Human activity recognition with two body-worn accelerometer sensors. Masters Thesis, Norwegian University of Science and Technology, Trondheim, Norway.

Hirshkowitz, M., 2015. The history of polysomnography: Tool of scientific discovery. Sleep Medicine: A Comprehensive Guide to Its Development, Clinical Milestones, and Advances in Treatment, 91–100.

Khademi, A., EL-Manzalawy, Y., Buxton, O. M., Honavar, V., 2018. Toward personalized sleep-wake prediction from actigraph. IEEE EMBS International Conference on Biomedical and Health Informatics (BHI), 414–417.

Kim, M. J., Lee, G.-H., Kim, C.-S., Kim, W. S., Chung, Y.-S., Chung, S., Lee, S.-A., 2013. Comparison of three actigraphic algorithms used to evaluate sleep in patients with obstructive sleep apnea. Sleep Breath 17, 297–304.

Kräuchi, K., Cajochen, C., Werth, E., Wirz-Justice, A., 1999. Warm feet promote the rapid onset of sleep. Nature 401, 36–37.

Kräuchi, K., Cajochen, C., Werth, E., Wirz-Justice, A., 2000. Functional link between distal vasodilation and sleep-onset latency? American Journal of Physiology 278 (3), R741–R748.

Lamprecht, M. L., Bradley, A. P., Tran, T., Boynton, A., Terrill, P. I., 2015. Multisite accelerometry for sleep and wake classification in children. Institute of Physics and Engineering in Medicine. Physiological Measurement 36 (1), 133–147.

Li, X., Zhang, Y., Sun, W., Song, Y., Dong, S., Lin, Q., Zhu, Q., Jiang, F., Zhao, H., 2018. A hidden markov model based unsupervised algorithm for sleep/wake identification using actigraphy. Submitted on 03 December 2018, arXiv:1812.00553 [stat.AP].

Neves, E. B., Vilaça-Alves, J., Antunes, N., Felisberto, I. M., Rosa, C., Reis, V. M., 2015. Different responses of the skin temperature to physical exercise: Systematic review. 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 1307–1310. doi: 10.1109/EMBC.2015.7318608.

Orellana, G., Held, C. M., Estevez, P. A., Perez, C. A., Reyes, S., Algarin, C., Peirano., P., 2014. A balanced sleep/wakefulness classification method based on actigraphic data in adolescents. Conf Proc IEEE Eng Med Biol Soc.2014;2014:4188-4191.

Phan, H., Andreotti, F., Cooray, N., Oliver Y. Chén, S. M., Vos, M. D., 2019. Joint classification and prediction cnn framework for automatic sleep stage classication. . Published in IEEE Transactions on Biomedical Engineering, arXiv:1805.06546 [cs.LG].

Picchietti, D., Winkelman, J. W., 2005. Restless legs syndrome, periodic limb movements in sleep, and depression. Sleep 28 (7), 891–898.

Raymann, R. J., Swaab, D. F., Someren, E. J. V., 2007. Skin temperature and sleep-onset latency: Changes with age and insomnia. Physiology & Behavior 90, 257–266.

Reinsve, Ø., 2018. Data analytics for hunt: Recognition of physical activity on sensor data streams. Masters Thesis, Norwegian University of Science and Technology, Trondheim, Norway.

Sadeh, A., Sharkey, M., Carskadon, M. A., 1994. Activity-based sleep-wake identification: An empirical test of methodological issues. Sleep 17 (3), 201–207, https://doi.org/10.1093/sleep/17.3.201.

Slates, J. A., Botsis, T., Walsh, J., King, S., Straker, L. M., Eastwood, P. R., 2015. Assessing sleep using hip and wrist actigraphy. Sleep and Biological Rhythms 13 (2), 172–180.

Tan, P.-N., Steinbach, M., Kumar, V., 2014. Introduction to Data Mining, 1st Edition. Pearson Education Limited, international Edition.

Tang, Y., Zhang, Y.-Q., Chawla, N. V., Krasser, S., 2009. Svms modeling for highly imbalanced classication. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39 (1), 281 – 288.

Tilmanne, J., Urbain, J., Kothare, M. V., Wouwer, A. V., Kothare, S. V., 2008. Algorithms for sleepwake identication using actigraphy: a comparative study and new results. Journal of Sleep Research 18, 85–98.

Tudor-Locke, C., Barreira, T. V., Jr., J. M. S., Mire, E. F., Katzmarzyk, P. T., 2013. Fully automated waist-worn accelerometer algorithm for detecting childrens sleep-period time separate from 24-h physical activity or sedentary behaviors. Applied Physiology, Nutrition, and Metabolism 39 (1), 53–57.

Vågeskar, E., 2017. Activity recognition for stroke patients. Masters Thesis, Norwegian University of Science and Technology, Trondheim, Norway.

van Hees, V. T., Sabia, S., Jones, S. E., Wood, A. R., Anderson, K. N., Kivimaki, M., Frayling, T. M., Pack, A. I., Bucan, M., Mazzotti, D. R., Gehrman, P. R., Singh-Manoux, A., Weedon, M. N., 2018. Estimating sleep parameters using an accelerometer without sleep diary. Scientific Reportsvolume 8, article number:12975, DOI:10.1038/s41598-018-31266-z.

Wang, P., Ji, L., Yan, J., Dou, D., de Silva, N., Zhang, Y., Jin, L., 2018. Concept and attention-based cnn for question retrieval in multi-view learning. ACM Transactions on Intelligent Systems and Technology 9 (4).

Xu, C., Tao, D., Xu, C., 2013. A survey on multi-view learning. Neural Computing and Application. arXiv:1304.5634 [cs.LG].

Xu, Z., Sun, S., 2010. An algorithm on multi-view adaboost. In: Wong K.W., Mendis B.S.U., Bouzerdoum A. (eds) Neural Information Processing. Theory and Algorithms. ICONIP 2010.Lecture Notes in Computer Science, vol 6443. Springer, Berlin, Heidelberg.

Yeo, M., Koo, Y. S., , Park, C., 2017. Automatic detection of sleep stages based on accelerometer signals from a wristband. IEIE Transactions on Smart Processing and Computing 6 (1), 21–26.

Zinkhan, M., Berger, K., Hense, S., Nagel, M., Obst, A., Koch, B., Penzel, T., Fietze, I., Ahrens, W., Young, P., Happe, S., Kantelhardt, J. W., Kluttig, A., Schmidt-Pokrzywniak, A., Pillmann, F., Stang, A., 2014. Agreement of different methods for

assessing sleep characteristics: a comparison of two actigraphs, wrist and hip placement, and self-report with polysomnography. Sleep Medicine 15 (9), 1107–1114.

Appendix A

Literature Review

This appendix contains additional information connected to how we performed our structured literature review.

A.1 Search Terms

The final selection of search terms used during our literature review can be viewed in Table A.1.

	Group 1	**Group 2**	**Group 3**
Term 1	Sleep analysis	Actigraphy	Algorithm
Term 2	Sleep patterns	Actimetry sensor	Method
Term 3	Sleep-states	Accelerometer	
Term 4	Sleep study	Actigraph	
Term 5	Sleep-wake patterns		
Term 6	Sleep detection		

Table A.1: Final selection of search terms

The search strategy was to implement AND, and OR statements between the terms. E.g, if you had three groups each with three search terms the complete sentence used for searching literature would be as follows:

$$([Group1, Term1]OR[Group1, Term2]OR[Group1, Term3])AND$$
$$([Group2, Term1]OR[Group2, Term2]OR[Group2, Term3])AND$$
$$([Group3, Term1]OR[Group3, Term2]OR[Group3, Term3])$$

For each round of searching only the first hundred search results from each source was gone through and evaluated as relevant or not.

A.2 Quality Assessment

The following questions was used to assess the quality of each paper contained in the initial selection of literature.
Quality Criteria:

- QC 1 Is there is a clear statement of the aim of the research?
- QC 2 Is the study is put into context of other studies and research?
- QC 3 Are system or algorithmic design decisions justified?
- QC 4 Is the test data set reproducible?
- QC 5 Is the study algorithm reproducible?
- QC 6 Is the experimental procedure thoroughly explained and reproducible?
- QC 7 Is it clearly stated in the study which other algorithms the study's algorithm(s) have been compared with?
- QC 8 Are the performance metrics used in the study explained and justified?
- QC 9 Are the test results thoroughly analysed?
- QC 10 Does the test evidence support the findings presented?

If the answer to a question was yes the paper was given a score of 1 for that question. If the answer was no the score would be 0. Sometimes the answer was "Partially". In that case the score given would be 0,5. It was decided that any paper with a total score lower than 7 would be discarded from the literature selection and not used any further.

Appendix B

Arousal Classification

This appendix contains some additional information with regards to our arousal (multi-class) experiments.

B.1 Arousal Types

The different arousal types found in the obtained arousal annotations for the PL data set is listed as follows:

- Arousal
- Arousal (Autonomic)
- Arousal (EEG)
- Arousal (EMG)
- Cardiac Arousal (Autonomic)
- Cardiac Arousal (EEG)
- Cardiac Arousal (EMG)
- Flow Limitation Arousal (EMG)
- LM Arousal
- LM Arousal (Autonomic)
- LM Arousal (EEG)

- LM Arousal (EMG)
- PLM Arousal (Autonomic)
- PLM Arousal (EEG)
- PLM Arousal (EMG)
- Respiratory Arousal (Autonomic)
- Respiratory Arousal (EEG)
- Respiratory Arousal (EMG)
- Snore Arousal (Autonomic)
- Snore Arousal (EEG)
- Snore Arousal (EMG)
- SpO2 Arousal (Autonomic)
- SpO2 Arousal (EEG)
- SpO2 Arousal (EMG)

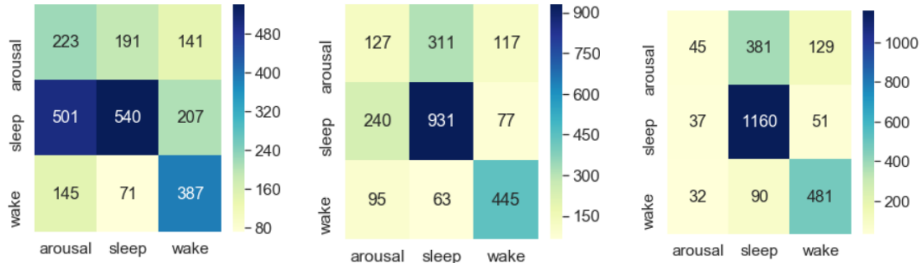
Appendix C

Results

This appendix contains some additional overviews of results not already included in Chapter 7.

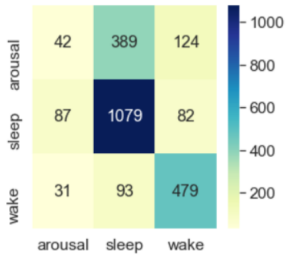
C.1 Confusion Matrices - Multiclass Classification

This section presents some of the confusion matrices resulting from our experiments with multiclass classification. The resulting confusion matrices after training and testing with all arousal types can be seen in Figure C.1. The resulting confusion matrices after training and testing XGB with feature selection is shown in Figure C.2. Figure C.3 and Figure C.4 presents the resulting confusion matrices after training and testing with only PLM arousals and with only balanced PLM arousals, respectively.

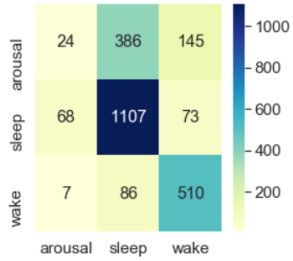


(a) DT **(b)** RF **(c)** XGB

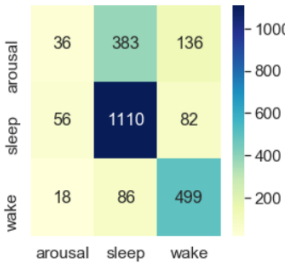
Figure C.1: Confusion matrices - Multiclass classification with all arousal types.



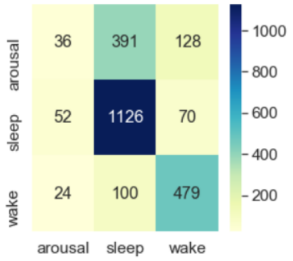
(a) VarianceThreshold



(b) SelectKBest

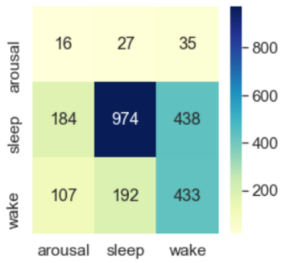


(c) L1 Based

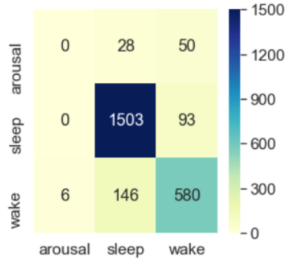


(d) TreeBased

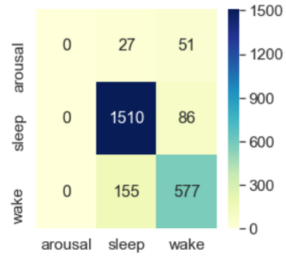
Figure C.2: Confusion matrices - Multiclass XGB with feature selection



(a) DT

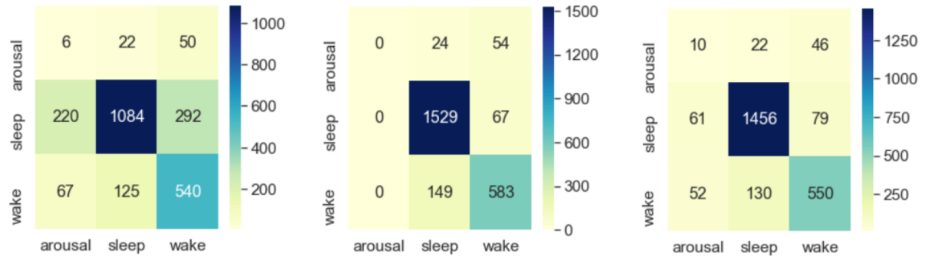


(b) RF



(c) XGB

Figure C.3: Confusion matrices - Multiclass classification with PLM arousals.



(a) DT **(b)** RF **(c)** XGB

Figure C.4: Confusion matrices - Multiclass classification with balanced PLM arousals.

C.2 Confusion Matrices- SuMV with K-means Clustering

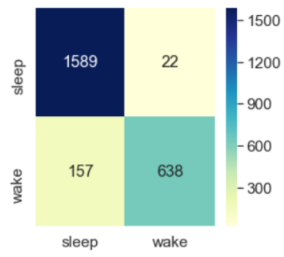
This section presents some of the confusion matrices resulting from our experiments with our supervised multiview with k-means clustering method.

Figure C.5 presents the resulting confusion matrices using SuMV with K-means clustering with our standard training and test sets. Figures C.6, C.7, C.8, and C.9 presents the resulting confusion matrices after using SuMV with K-means clustering and data sets obtained through the CoSV and CoMV methods as training data.

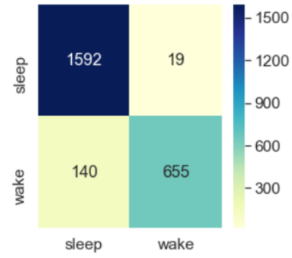
C.3 Confusion Matrices - SuMV with Agglomerative Hierarchical Clustering

This section presents some of the confusion matrices resulting from our experiments with our supervised multiview with agglomerative hierarchical clustering (AHC) method.

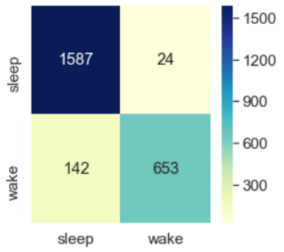
Figure C.10 presents the resulting confusion matrices using SuMV with AHC with our standard training and test sets. Figures C.11, C.12, C.13, and C.14 presents the resulting confusion matrices after using SuMV with AHC and data sets obtained through the CoSV and CoMV methods as training data.



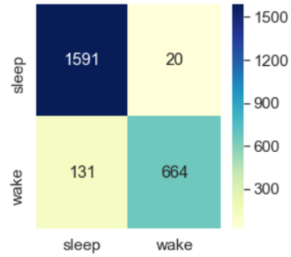
(a) DT



(b) RF

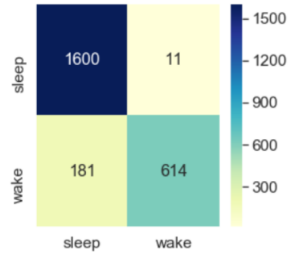


(c) XGB

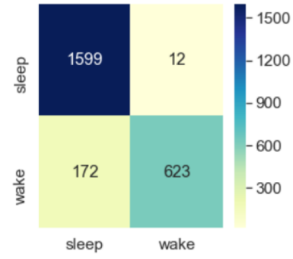


(d) All

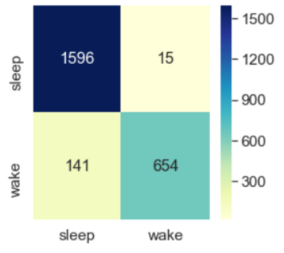
Figure C.5: Confusion matrices - Supervised Multi-view with k-Means clustering



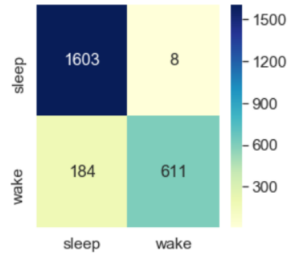
(a) DT



(b) RF

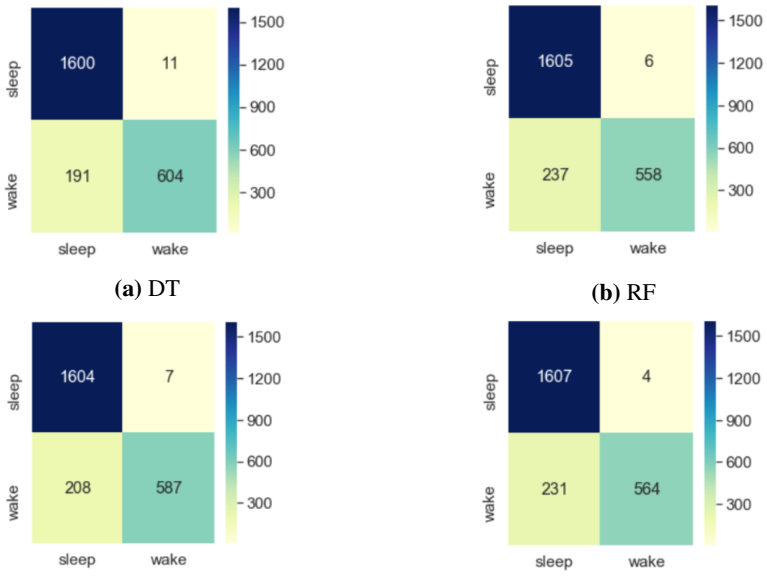


(c) XGB



(d) All

Figure C.6: Confusion matrices - Supervised Multi-view with k-means clustering using CoSV data set



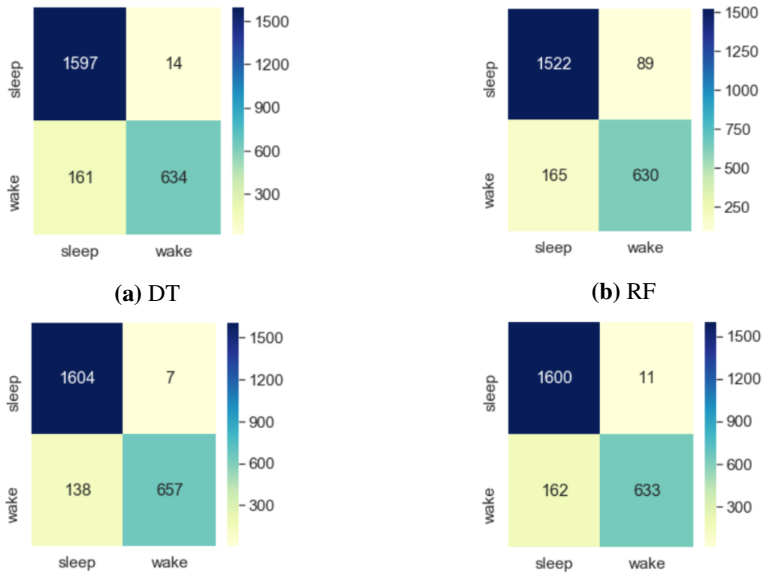
(a) DT

(b) RF

(c) XGB

(d) All

Figure C.7: Confusion matrices - Supervised Multi-view with k-means clustering using CoMV-DT data set



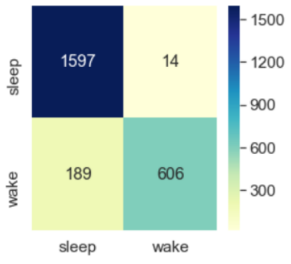
(a) DT

(b) RF

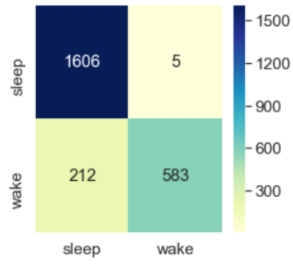
(c) XGB

(d) All

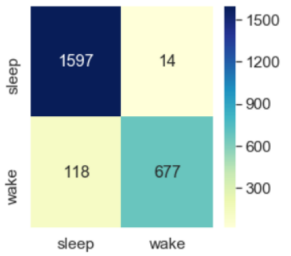
Figure C.8: Confusion matrices - Supervised Multi-view with k-means clustering using CoMV-RF data set



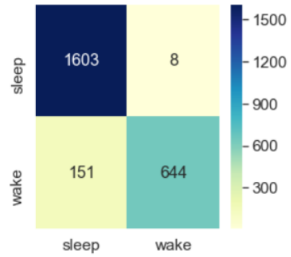
(a) DT



(b) RF

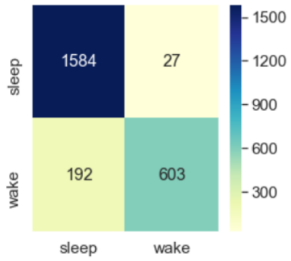


(c) XGB

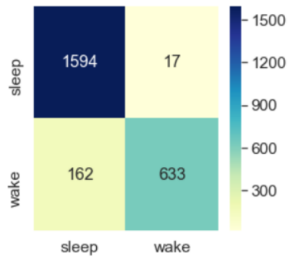


(d) All

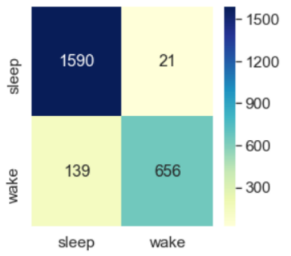
Figure C.9: Confusion matrices - Supervised Multi-view with k-means clustering using CoMV-XGB data set



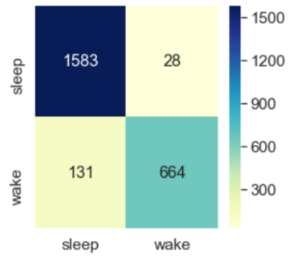
(a) DT



(b) RF

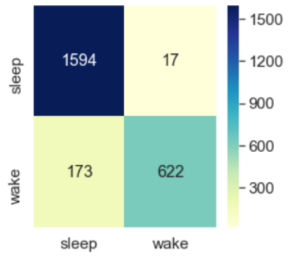


(c) XGB

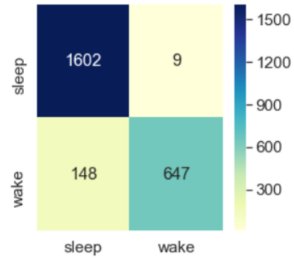


(d) All

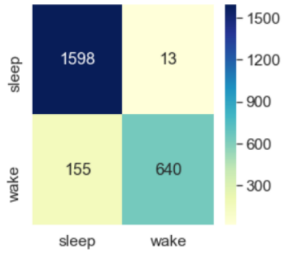
Figure C.10: Confusion matrices - Supervised Multi-view with agglomerative clustering



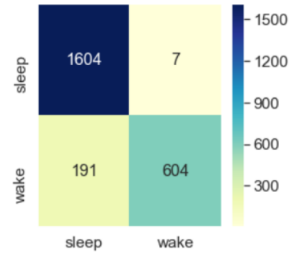
(a) DT



(b) RF

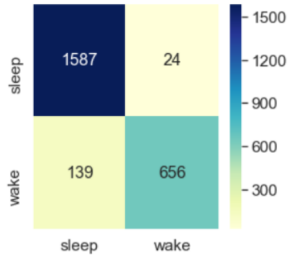


(c) XGB

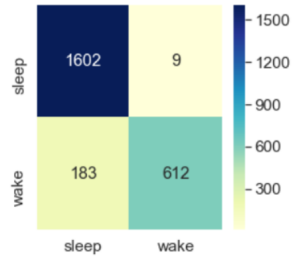


(d) All

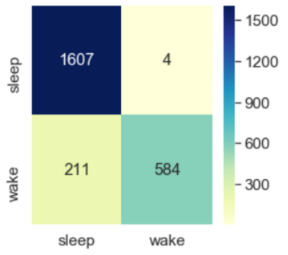
Figure C.11: Confusion matrices - Supervised Multi-view with agglomerative hierarchical clustering using CoSV data set



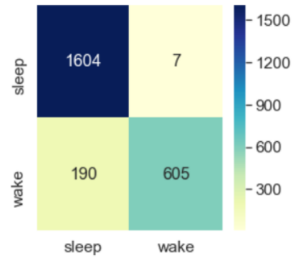
(a) DT



(b) RF

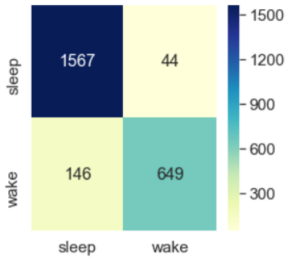


(c) XGB

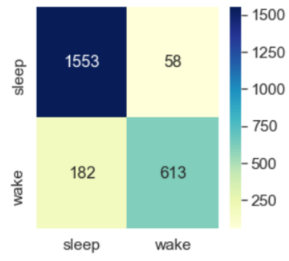


(d) All

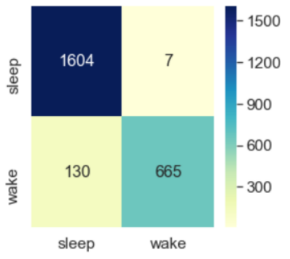
Figure C.12: Confusion matrices - Supervised Multi-view with agglomerative hierarchical clustering using CoMV-DT data set



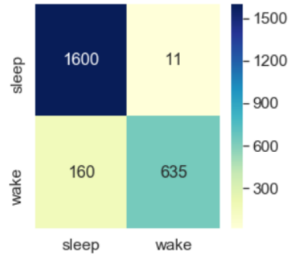
(a) DT



(b) RF

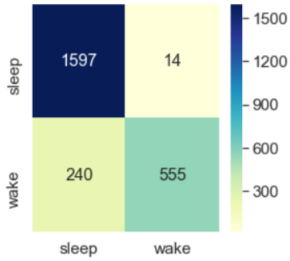


(c) XGB

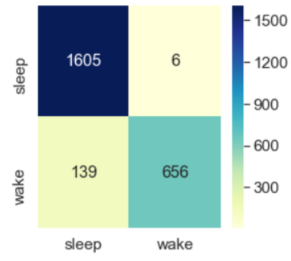


(d) All

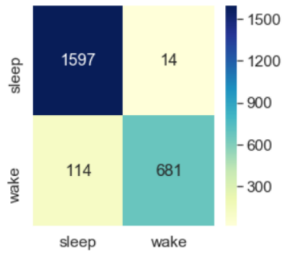
Figure C.13: Confusion matrices - Supervised Multi-view with agglomerative hierarchical clustering using CoMV-RF data set



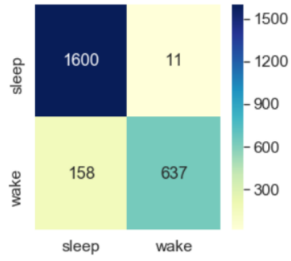
(a) DT



(b) RF



(c) XGB



(d) All

Figure C.14: Confusion matrices - Supervised Multi-view with agglomerative hierarchical clustering using CoMV-XGB data set