Masteroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for teknisk kybernetikk

Julie Haga

Biometric system using EEG signals from resting-state and one-class classifiers

Masteroppgave i Kybernetikk og robotikk
Veileder: Marta Molinas
Juni 2020

NTNU
Norwegian University of
Science and Technology

Julie Haga

Biometric system using EEG signals from resting-state and one-class classifiers

Masteroppgave i Kybernetikk og robotikk
Veileder: Marta Molinas
Juni 2020

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for teknisk kybernetikk

NTNU

Kunnskap for en bedre verden

NTNU – Trondheim
Norwegian University of
Science and Technology

Biometric system using EEG signals from resting-state and one-class classifiers

Julie Haga

Submission date: Monday 1st June, 2020
Supervisor: Marta Molinas
2^{nd} Supervisor: Luis Alfredo Moctezuma

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Abstract

In this thesis, a Electroencephalography (EEG)-based biometric system is implemented. The goal is to investigate the possibility of authenticating subjects based on EEG signals.

A python application has been designed and implemented to realize the real-time system. Several approaches for the application is proposed, and different experiments are designed to investigate the potential of these methods.

The EEG data used in this work is taken from two different public databases. The first dataset contains data recorded from 26 subjects using a P300-speller system, preforming five sessions consisting of 60 trials. The data was recorded using 56 channels. The second dataset is EEG-signals from 40 subjects, recorded with 64 channels. The subjects performed eight sessions with 24 trials each.

The methods used for feature extraction are Discrete Wavelet Transform (DWT), Principal Component Analysis (PCA) and Empirical Mode Decomposition (EMD). Additionally, energy and fractal features have been extracted from the decomposed signals. The classifiers used in the experiments are One-class Support Vector Machine (OC SVM) and autoencoders built of Convolutional Neural Network (CNN). An effort has been made to investigate if the performance can be maintained by reducing the number of channels used for recording. The channels are selected using a Genetic Algorithm (GA). Additionally, the GA is used to find optimal hyperparameters for the OC SVM.

A model created using 24 instances was able to authenticate 40 subjects with an True Acceptance Rate (TAR) and True Rejection Rate (TRR) of 0.96 and 0.94. This result was obtained using a CNN autoencoder and 64 channels. Experiments with reduced training data are constructed to improve real-time aspects. The best result was obtained using only 2 channels and a CNN autoencoder with single-channel convolution. Then the TAR was 0.97 and TRR 0.95 for 40 subjects. Using only 18 training instances on 20 subjects yield 1.0 for both TAR and TRR.

Sammendrag

I denne masteroppgaven er et EEG-basert biometrisk system implentert. Målet er å undersøke muligheten for å authentisere personer basert på EEG signaler. Systemet er realiesert i sanntid gjennom en python applikasjon. En rekke metoder er foreslått for designet av applikasjonen og ulike eksperimenter er designet for å teste potensialet til disse metodene.

EEG-dataen som brukes i dette arbeidet er hentet fra to offentlig databaser. Det første datasettet inneholder data registrert fra 26 deltagere ved bruk av et P300-stavingssystem, hvor hver deltaker gjennomfører 5 økter bestående av 60 forsøk. Dataen er registrert med 56 sensorer. Det andre datasettet er EEG-signaler fra 40 deltagere, registert med 64 sensorer. Deltagerene gjennomførte åtte økter med 24 forsøk hver.

Metodene som brukes for å finne karakteristiske element er DWT, PCA og EMD. I tillegg hentes energiske og fraktale elementer ut fra signalene. Klassifikasjonsmetodene som er testet i eksperimentene er OC SVM og autoencodere bygget av CNN. Det er undersøkt om ytelsen i systemet kan opprettholdes når antall sensorer reduseres. Sensorene velges ved å bruke en GA. I tillegg brukes GA for å finne optimale hyperparametere for OC SVM.

En modell som er trent på 24 forsøk fra hver deltaker kunne autentisere 40 personer med en TAR og TRR på henholdsvis 0.96 og 0.94. Dette resultatet ble oppnådd ved bruk av en CNN autoencoder og 64 sensorer. Eksperimenter med redusert treningsdata er gjennomført for å forbedre ytelsen i sanntid. Det beste resultatet ble oppnådd ved å bruke kun 2 sensorer og en autoenkoder med singel-sensor konvolusjon. Da var TAR 0.97 og TRR 0.95 for 40 personer. Eksperiment på 20 personer med 18 forsøk for trening ga TAR og TRR lik 1.0.

Acknowledgements

This master thesis is submitted as the final part of my master's degree at the Department of Engineering Cybernetics at the Norwegian University of Science and Technology.

The project is supervised by Professor M. Molinas, whom I want to thank for introducing me to the topic, and for forming the idea and baseline for this project.

Also, I want to express enormous gratitude to Luis Alfredo Moctezuma, for his programming guidance, and valuable feedback on my thesis. You have been an incredible resource throughout the last couple of semesters.

Thanks to Shobiba Premkumar for great collaboration on the implementation of the system. Not least, for helpful discussions along the way.

Not to mention, thanks to my parents for welcoming me home and letting me station myself at the home office when campus closed due to the pandemic.

Finally, I would like to thank all my friends in Trondheim; it would not have been the same without you.

Julie Haga

Trondheim, June 1st 2020

Contents

List of Figures

List of Tables

List of Acronyms

ANN Artificial Neural Network.

AR Autoregressive.

BCI Brain Computer Interface.

CNN Convolutional Neural Network.

DL Deep Learning.

DT Decision Tree.

DWT Discrete Wavelet Transform.

EEG Electroencephalography.

EMD Empirical Mode Decomposition.

ERP Event-related Potentials.

FDA Fisher's Discriminant Analysis.

FFT Fast Fourier Transform.

FPA Flower Pollination Algorithm.

GA Genetic Algorithm.

HFD Higuchi Fractal Dimension.

IE Instantaneous Energy.

IMF Intrinsic Mode Functions.

IQR Interquartile range.

k-NN k-Nearest Neighbors.

MI Mutual Information.

ML Machine Learning.

MOOP Multi-Objective Optimization Problem.

MSE Mean Square Error.

OC SVM One-class Support Vector Machine.

ORM Object-Relational Mapping.

PAL Part Average Limit.

PC Principal Components.

PCA Principal Component Analysis.

PFD Petrosian Fractal Dimension.

PSD Power Spectral Density.

RBF Radial Basis Function.

SVM Support Vector Machine.

TAR True Acceptance Rate.

TE Teager Wavelet Energy.

TRR True Rejection Rate.

VEP Visual Evoked Potential.

WT Wavelet Transform.

Chapter 1
Introduction

As human interactions take more and more place in a digital context, the need for methods to prove our identity is rising. Not only do we need to secure persons, objects and data, but there is a growing demand to increase the *reliability* of the identity of persons. Traditional identification technologies, such as check identity documents and access system based on password authentication, are at their limits. To increase the level of security of identification contributes *biometric identification*. Besides, these technologies save time, create less hassle, reduce staff costs and maintain maximum efficiency.

Biometrics is a technique used for identifying subjects with unique human biological features like fingerprints, face, iris, and voice [1]. Biometrics is a popular research topic because reliable biometric systems are interesting to all facilities where a minimum of security access is required.

The interest in finding a new biometric mark for subject identification is increasing in correlation with the rising vulnerability in the existing systems. The biometrics used today is not *secret*; thus, they are vulnerable to security threats such as spoofing and masquerade attacks. Identity fraud is one of the more common criminal activities and is associated with high costs and severe security issues.

The following demands are defined for a living physical or behavioural trait to be used in a biometric application [2]. The trait must be *universal* (every individual should possess the trait), *unique* (the given trait should be sufficiently different across the population), *permanent* (it should not change over time) and *measurable* (it should be possible to acquire and digitise the biometric trait).

EEG is a technique used to record the electrical activity generated by the brain from electrodes placed on the scalp. As brainwave signals meet all of the conditions presented above, EEG stands as a strong candidate for a new biometric mark. The brainwave signals are confidential and extremely complex, which makes them hard

to steal, duplicate or falsify. Even more, brain signals are dependent on the mood and stress of the subject, making it very difficult to get them by force [3].

In addition to being reliable, an ideal biometric system must be user-friendly, fast and of low cost. Earlier studies on EEG-signals have demonstrated that they can be used for subject identification with high accuracy. One of the significant challenges and drawbacks of these experiments are the amount of data and the number of channels required for correct classification, which is an essential issue for real-time processing and costs.

In a biometric recognition system, we differentiate between the *authentication*, which confirms or denies an identity claim by a particular individual, and *identification*, which identifies an individual from a group of persons. The scope of this work is limited to the authentication process.

There are two types of access attempts for an authentication system. An *user attempt* (a user claims its real identity and should be accepted) and an *intruder attempt* (the user is not enrolled in the system and should be rejected). The performance of the biometric system against these attempts is measured by the TAR and TRR, respectively. The TAR is a statistic used to measure the performance of the user attempts. It is the percentage of times the system correctly accepts an enrolled user. The TRR is the percentage of times the system correctly rejects an intruder.

1.1 Problem description

Relying on the foundation that EEG-signals are unique for individuals, this work aims to design and implement a user-friendly biometric system that meets the time and mobility demands of a real-time application.

The complete system is implemented in collaboration with another student. However, this study is limited to address the authentication layer of the biometric system.

The study addresses a wide range of relevant topics, such as concepts within signal analysis and Machine Learning (ML), state-of-the-art research on the topic of authentication using EEG, and GA for solving Multi-Objective Optimization Problem (MOOP). These subjects are presented and discussed.

The problem is approached by testing several methods in every step of the authentication model. Different methods for feature extraction are explored; DWT, PCA and EMD. Also, different methods for classification, using both OC SVM and Artificial Neural Network (ANN). Additionally, different approaches for the overall authentication methodology is reviewed. The methods are inspired by the variety of

techniques investigated in similar studies. Several combinations will be examined in the search of finding the optimal approach.

Emphasis has been placed on user-friendly and real-time aspects. By reducing the number of channels, one can increase the system efficiency and mobility, as well as reduce the costs of equipment. Besides, reducing the data size used for training is an important issue, as shorter training time increases the efficiency of the system.

1.1.1 Research questions

The following research questions have been formulated for this work:

1. Is it possible to design a classifier that can separate between enrolled users and intruders based on EEG-signals?
2. Can the performance of the classifier be maintained by reducing the number of channels used for recording the signals?
3. May such a classifier meet the requirements for a real-time application?

1.1.2 Motivation

Biometric system based on EEG is a topic addressed by several researchers. However, several aspects need to be improved before the commercialisation of such a system. This work efforts towards finding a suitable approach for the realisation of a user-friendly biometric system that can operate in real-time.

1.2 Report structure

A selection of background theory related to EEG, signal analysis, ML and optimisation problems is described in chapter 2. Subsequently, the report gives a summary of the state-of-the-art to related topics in the context of authentication based on EEG-signals, such as protocols, feature extraction, channel selection, and classification. Also, an overview of the use of Deep Learning (DL) and EEG is presented. In chapter 4, the complete approach used for system design, signal analysis, feature extraction, classification and software implementation is given in detail. The chapter also provides a description of the datasets that are used. Experiments are designed to test the potential of the proposed methods; the result of these experiments are presented in chapter 5. The methods and results are discussed throughout the chapter. Finally, a conclusion and suggestions for future work is given in chapter 6.

Chapter 2
Background

The objective of this chapter [1] is to provide background knowledge considered useful for experiments related to EEG-signals. The first section addresses topics related to the human brain, followed by a section presenting the data processing methods used in this work and material related to ML. Finally, some theory related to optimisation problems and GA is given.

2.1 The human brain

2.1.1 Structure of the human brain

The brain's cerebral cortex is divided into the left and right cerebral hemispheres; these again are divided into four lobes (frontal, parietal, temporal, and occipital). Most brain functions activate different regions of the brain, but some functionalities are peculiar to specific lobes. The frontal lobe is associated with reasoning, motor skills, higher-level cognition, and expressive language. Processing of body senses occurs in the parietal lobe. The temporal lobe is the main area for cognitive functions such as memory, speech, and language skills. Processes regarding visual stimuli, recognizing objects, and identifying colours appear in the occipital lobe [5].

2.1.2 Electroencephalography

EEG is a technique utilized for recording the electrical activity generated by the brain. All data in this work is from noninvasive EEG, as the electrodes are placed on the scalp for recording. EEG measures voltage fluctuations resulting from ionic current within the neurons of the brain [6].

The electrical activity is due to changes in the membrane potential in a neuron. Ions are pushed across the cell membrane by ion pumps. This activity causes an electrical

[1] This chapter is an extended version of the background chapter presented in the author's work presented in [4].

potential across the cell membrane called the *resting potential*. A nervous signal triggers an *action potential*, which is a depolarizing of the cell [7].

The electric potential generated by an individual neuron is far too small to be picked up by EEG. The recorded waveforms reflect the summation of the synchronous activity of thousands or millions of neurons [7].

2.1.3 Electrode placement

Scalps electrodes are used to record the EEG-signals. These are usually placed according to the international 10-20 system [8]. Each site has a letter to identify the brain lobe. The lobes of a human brain are described in section 2.1.1. The letters F, P, T, O, and C stands for Frontal, Parietal, Temporal, Occipital, and Central (there is no central lobe, this is just for identification purpose). When using more electrodes, some electrodes are placed in intermediate sites. These placements are denoted with two letters. For instance, TP refers to the site between the temporal and parietal lobe.

The right and the left hemisphere are referred to by even and odd numbers, respectively. The numbers 10 and 20 tells if the distance between adjacent electrodes is either 10% or 20% of the total front-back or right-left distance of the scalp. The placement methodology is visualized in fig. 2.1 [2].

The electrodes are placed on different types of devices, such as helmets, caps and headsets. The main difference between these devices is the number of electrodes used for recording.

2.1.4 Brain rythms

Brain waves have been grouped according to their frequencies, referred to as the frequency bands of the brain [3]. The different frequency bands are given together with their associated mental state in table 2.1.

Table 2.1: Frequency bands of the brain [3].

Brain rhythm	**Frequency**	**Associated with**
Delta wave (δ)	0.5 - 4Hz	Deep sleep
Theta wave (θ)	4 - 8Hz	Day dreaming and meditation
Alpha wave (α)	8 - 12Hz	Awake, but relaxed
Beta wave (β)	12 - 30Hz	Awake, and thinking
Gamma wave (γ)	$>$ 30 Hz	Deep focus

[2]The image is used with written consent from Trans Cranial Technologies.

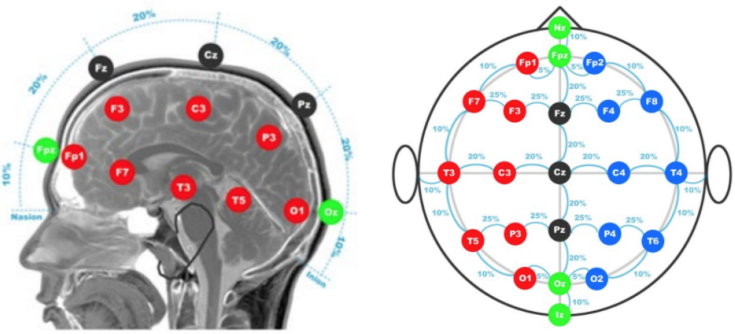

Figure 2.1: Placement of the electrodes according to the international 10-20 system [9].

2.1.5 Event-Related Potential

EEG is typically collected when the subject engages in a specific activity. The protocols for recording can be separated into two categories: resting-state/idle-state and cognitive tasks. Resting-state EEG is acquired when the participant is simply at rest. These protocols are fairly simple and is therefore very common in brain biometrics [10].

The cognitive protocols are more complex. When using a cognitive protocol, the systems does not use the raw EEG time series directly, but the Event-related Potentials (ERP). ERP is the measured brain response that is the result of a specific event. The event can be a motoric event (button press, eye movement), a mental operation (motor imagery), or a sensory event (flash of light, noise) [11]. To isolate the ERP, many trials must be conducted and averaged, which causes random brain activity to be cancelled out.

Visual Evoked Potential (VEP) is an ERP that is evoked by visual stimuli. It reflects the visual information-processing mechanism in the brain. Both VEPs and ERPs are usually easiest detected from the occipital lobe, where the processing of visual stimuli takes place [12], as described in section 2.1.1.

ERP in humans can be divided into 2 categories: sensory and cognitive. The sensory waves peaks within the first 100 milliseconds after stimulus. Cognitive waves are ERPs that generates later and reflects the subjects reaction to the stimulus or the subjects information processing. Waveforms are described according to their latency and amplitude. The capital letters P and N are used to determine whether the peak is postitive (P) or negative (N). This is followed by a number which indicates

the average peak latency. For example, P100 is a wave with a positive peak at approximately 100 ms following stimulus onset. Other example waves are N100, P200, N200 and P300, visualized in fig. 2.2.

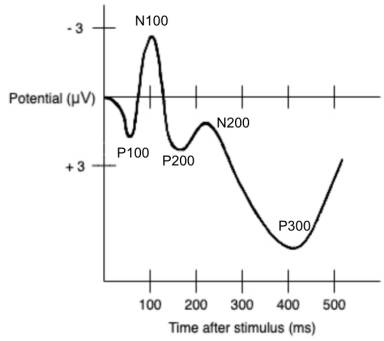

Figure 2.2: ERP waveforms

2.2 Data preprocessing and feature extraction

Since its starting point in 1929, EEG has been interpreted by visual inspection of waveforms. Particularly, in the field of medicine, visual inspection has provided the basis for many findings such as dyslexia and epilepsy [13]. By using feature extraction and computer-assisted analysis, one can decrease the complexity of the EEG-signals, and the information in the signals are more accessible. Also, by reducing the complexity, one can increase the accuracy of detection.

The objective of feature extraction is to describe the signals in terms of a small number of relevant variables. This stage is essential for processing and analysis of the EEG-signal because the extracted features influence the performance of the recognition system [10].

Features can be extracted from the different domains, such as time-domain, frequency-domain or time-frequency-domain. They can be extracted directly from the raw signal or after the raw signal is processed. The following section gives a theoretical description of the data processing techniques and features used in this work.

2.2.1 Wavelet Transform

The Wavelet Transform (WT) provides a time-frequency representation of a signal. The EEG signal is of nature time-invariant. Hence, a time-frequency representation of the signal is useful.

WT decomposes a signal in the time-domain into shifted and scaled versions of a base wavelet, called the mother wavelet. Some examples of mother wavelets are Morlet, Mexican hat, Biorthogonal and Symlet. A challenge when using the WT is selecting the optimal mother wavelet, as various wavelets applied on the signal may produce different results. It is common to select a mother wavelet that is similar in shape to the original signal. However, when using a complex signal such as the EEG, visual similarities can be challenging to find.

The method divides the signal into shorter segments and transforms each of the segments separately. The WT enables variable window sizes in analyzing different frequency components within a signal [14]. For high frequencies, a short duration function is used, while a longer duration is applied for low frequencies. In this way, WT provides a flexible resolution in both time and frequency.

In the DWT, a signal is high-pass and low-pass filtered, producing high- and low-pass sub-bands. In fig. 2.3, a schematic overview of the algorithm is presented. The mother wavelet is high-pass in nature; thus, it constitutes the first high-pass filter. Its mirrored version is low-pass and corresponds to the first low-pass filter. The outputs provide the level 1 high-frequency part named detail coefficients (D1), and the level 1 low-frequency part, named approximation coefficients (A1). Subsequently, the low pass portion is fed into a new set of filters. This process is repeated until the signal is decomposed to a pre-defined level [15]. The result is a set of sub-bands, each covering a frequency range. At every level, half of the samples can be eliminated according to the Nyquist's rule [16]. The procedure described is known as the multi-resolution decomposition of a signal.

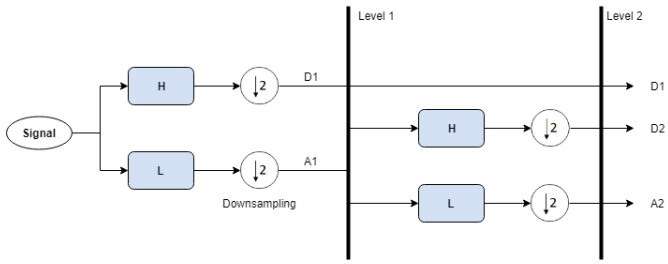

Figure 2.3: Flowchart of the sub-band coding algorithm, L and H indicates low and high-pass filters respectively.

2.2.2 Principal Component Analysis

PCA is a common method for feature extraction and dimensional reduction that uses orthogonal transformation to convert a set of correlated variables into a set of

Principal Components (PC)s. PCA is based on a decomposition of the raw data matrix $\mathbf{X}$ into two matrices $\mathbf{V}$ and $\mathbf{U}$, i.e $\mathbf{X} = \mathbf{U} \cdot \mathbf{V}^T$.

The matrices $\mathbf{V}$ and $\mathbf{U}$ are orthogonal. The $\mathbf{V}$ are the weights for each original variable from the data when calculating the PCs; it is called the loading matrix. The $\mathbf{U}$ matrix contains the original data in the new coordinate system and is called the score matrix.

The PCs are the underlying structure in the data. These components are found by calculating the covariance matrix $\mathbf{X}$ of the data points. Covariance determines the relationship between the movement of two variables. From the covariance matrix, the eigenvectors and the corresponding eigenvalues are calculated. Eigenvectors and eigenvalues exist in pairs: the eigenvector gives the direction, and the eigenvalues address the variance in the data in that direction. The eigenvectors of a matrix are always orthogonal, hence linearly independent. These vectors form a new basis for the original data. Dependent on the dimension wanted, n PCs are chosen to form the feature vector [17].

The eigenvectors are ranked according to their eigenvalues (variance) in decreasing order. Hence, the first few principal components contain the most information about the original data. By ignoring the less important components, the data dimension can be reduced [10].

2.2.3 Empirical Mode Decomposition

EMD is another algorithm used to decompose time-series data. The algorithm decomposes the signal into several Intrinsic Mode Functions (IMF)s. An IMF is defined as a function that satisfies the following requirements [18]:

- The number of local minima and maxima differs at most by one.
- The mean value of upper and lower envelopes equal to zero.

The process of extracting the IMFs from a time-series is called *Sifting* [18] and is described in algorithm 1. This method is entirely data-driven, which motivates for utilization of EMD on EEG-signals.

Cubic Spline is the most commonly used method to find the signal envelopes, and this method is also used for this work. However, the envelope obtained by the Cubic Spline Interpolation is prone to under- and overshooting, which means finding inaccurate extrema. This may cause an inaccurate decomposition as wrong extrema will lead to a wrong representation of the envelope area. Under- and overshooting

Algorithm 1: The sifting process for a signal $x(t)$

Data: Time serie = $x(t)$
Result: IMFs
sifting = True;
while *sifting = True* **do**
1. Identify all upper extrema in $x(t)$
2. Interpolate the local maxima to form an upper envelope $u(x)$.
3. Identify all lower extrema of $x(t)$
4. Interpolate the local minima to form an lower envelope $l(x)$
5. Calculate the mean envelope:
   $m(t) = \frac{u(x)+l(x)}{2}$
6. Extract the mean from the signal:
   $h(t) = x(t) - m(t)$

if $h(t)$ *satisfies the two IMF conditions* **then**
$h(t)$ is an IMF;
sifting = False ; ▷ Stop sifting
else
x(t)= h(t);
sifting = True ; ▷ Keep sifting
if $x(t)$ *is not monotonic* **then**
Continue;
else
Break;

is mostly due to the property of keeping smooth when the extrema are distributed unevenly. Also, because the curve is globally controlled, an outlier among the extrema will change the shape of the envelope. The extrema reflect not only the envelope shape but also the frequency components of the signal. So, under- and overshooting is the main reason to cause mode mixing in EMD. Mode mixing refers to the problem when an IMF contain signal with different scales or that similar scales exist in several IMFs [19].

2.2.4 Feature extraction

A variety of features can be extracted from the decomposition obtained with DWT or EMD, such as statistical values, several energies and entropy values. For this work, energy features and fractal features are used.

Energy features

Energy features provide information regarding instantaneous frequency and amplitude.

Instantaneous Energy (IE) reflects the amplitude of the signal and is computed as in eq. (2.1).

$$IWE_j = log_{10}\left(\frac{1}{N_j}\sum_{r=1}^{N_j}(w_j(r))^2)\right) \tag{2.1}$$

Teager Wavelet Energy (TE) extracts the signal energy based on mechanical and physical considerations [20]. The calculation of TE is given in eq. (2.2).

$$TWE_j = log_{10}\left(\frac{1}{N_j}\sum_{r=1}^{N_j-1}\left|(w_j(r))^2 - w_j(r-1) * w_j(r+1)\right|\right) \tag{2.2}$$

In the equations above the w_j is the wavelet coefficients in the j-th decomposition level, and N_j is the number of samples in the j-th decomposition level.

Fractal features

A fractal is a shape that retains its structural detail despite scaling. The fractal dimension is represented by a single number (often a fraction) that can be used as a fundamental quantification of even the most complex shapes [21]. Hence, complex objects or functions, such as EEG signals can be described with the help of the fractal dimension. There are many methods used to calculate fractal dimensions. However, the widely accepted ones are Petrosian Fractal Dimension (PFD) and Higuchi Fractal Dimension (HFD), those are therefore chosen for this work.

The PFD is a fast estimation of the fractal dimension. However, this is the fractal dimension of a *binary sequence*. Since waveforms are analogue signals, a binary signal can be derived. The PFD of the derived binary sequence can then be calculated as

$$PFD = \frac{log_{10}(n)}{log_{10}(n) + log_{10}(\frac{n}{n+0.4N_\Delta})} \tag{2.3}$$

where n is the length of the sequence (number of points), and $N\Delta$ is the number of sign changes (number of dissimilar pairs) in the binary sequence generated [21].

The HFD is an important measure in biological and medical research [22]. The HFD is a nonlinear measure of a waveform in the time domain. Discrete signals can be written as a time series $x(1), x(2), ..., x(N)$. A new self-similar time series can be calculated as:

$$X_k^m : x(m), x(m+k), x(m+2k), ..., x(m+int[(N-k)/k]k]) \tag{2.4}$$

for m = 1, 2, .., k, where m is the initial time, k is the time interval, and int(r) is the integer part of the real number r. The length of the curve $L_m(k)$ can then be computed for each of the k time series or curves X_k^m, as in eq. (2.5).

$$L_m(k) = \frac{1}{k}\left[\left(\sum_{i=1}^{int[\frac{(N-m)}{k}]} |x(m+ik) - x(m+(i-1)k|\right)\frac{N-1}{int[\frac{N-m}{k}]k}\right] \tag{2.5}$$

In eq. (2.5) N is the length of the time series and $(N-1)/int[(N-m)/k]k$ is a normalization factor. The mean of $L_m(k)$ is computed to find the HFD as

$$HFD = \frac{1}{k}\sum_{m=1}^{k} L_m(k). \tag{2.6}$$

2.3 Machine Learning

ML is the study of systems that can automatically learn and improve through experience, without being explicitly programmed [23]. The theory of statistics is used to build mathematical models based on *training data*. The models is then used in order to make predictions or decisions for unknown *test data*. ML is a broad topic with many sub-fields, the following sections will give a more detailed description of the concepts used in this work.

2.3.1 Basics of machine learning

Supervised and unsupervised learning

ML algorithms can be separated into supervised and unsupervised algorithms. The unsupervised algorithms are presented with a dataset and learn the structure that represents this data. For the supervised algorithms, labels are provided alongside the input data. Thus, the algorithms can learn the mapping from input to a specific label.

Model validation

Cross-validation is a model validation technique for estimating how the model will generalize to an independent data set [24]. Cross-validation tests the model's performance on data that was not used for creating the model by giving one dataset for training and another dataset for testing.

In k-fold cross-validation, k rounds of validation are performed to reduce variability. The objects in the test set are varied for each iteration, ensuring that the test set is

representative for the data set to be analyzed. The final result is averaged from all the rounds of cross-validation [24].

2.3.2 One-class Support Vector Machine

In a Support Vector Machine (SVM), the input data is represented in a N-dimensional space, where N is the number of features. The algorithm seeks to find a decision boundary or a hyperplane that can separate the data points into classes. The distance from each point to the decision boundary is called *support vectors*. The algorithm search for the decision boundary with *maximised margin*. That is the boundary that maximizes the sum of the support vectors [25].

In the case of one-class classification, this translate to identifying the smallest *hypersphere* (with radius r, and centre c) consisting of all the data points belonging to the class. The model is unsupervised; provided with only features. The model infers the properties of this class, and from these properties, the model can predict which examples from a test set that is different from the training examples.

How bad the classifier should avoid misclassifications in training is determined by the regularization parameter, C. There is a trade-off between the correct classification of training examples and maximization of the decision function's margin. For large values of C, all samples should be included by the decision boundary. However, if the training set includes outliers (instances that deviates significantly from the rest), these points should be discarded. As fig. 2.4 demonstrate, allowing for some misclassifications in training can result in better classification results for the test data. Outliers are taken into account by the hyperparameter *nu*, which sets the proportion of expected outliers in the training dataset.

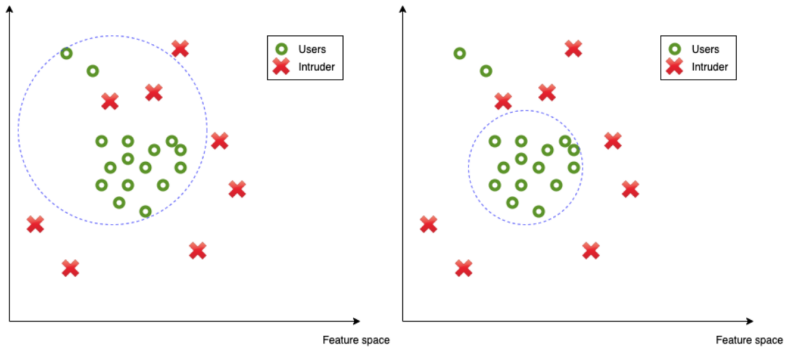

Figure 2.4: Two different decision boundaries for dataset with outliers.

Originally the SVM is meant for linearly separable classes. By projecting the data

through a non-linear function to a higher dimension space, it can create a non-linear decision boundary. This is called the *kernel trick* [26]. The Radial Basis Function (RBF) presented in eq. (2.7), is the most common kernel.

$$K(x_i, x_y) = exp(-\gamma||x_i - x_y||^2) \tag{2.7}$$

The algorithm uses the kernel function to find similarity between an unknown input x_y and the entire set of known instances x_i. As eq. (2.7) shows, the RBF goes to 1 when the instances are close, and 0 when they are far apart. Hence, the gamma value decides what is considered as close for the two points (same class) and far (different class). The challenges regarding this work are to find suitable nu and gamma values that will enclose all instances representing the users and not include any intruders.

2.3.3 Artificial Neural Networks

ANNs are inspired by the way the nervous system process information. It is composed of a large number of connected processing elements (neurons) that works in unison to solve a specific problem. ANN is the key component in DL, which teaches computers to learn by example. In DL, a computer model learns to perform classification tasks directly from images, text or sound [27]. An advantage with DL is that the system can learn feature levels with a minimum of human contribution. The drawback is that deep networks are large and demands much memory and high computation power. The theory presented in the next paragraphs is collected from the material presented in [28].

Components in ANN

The foundation of ANN is neurons, which takes an input and multiply it by the connected weights. One neuron can have multiple inputs, $x0, x1, x2, x3..x(n)$, which are independent variables that constitutes the input vector $\mathbf{x}$. Each input is multiplied by a connection *weight*, represented by $w0, w1, w2, w3...w(n)$. All products are summed up in the neuron. Mathematically this can be written $\sum_j w_j x_j$. The sum is then applied to an activation function ϕ. The activation function converts the input signal into an output signal which is used as input to the next layer. These components together add up a *perceptron*, shown in fig. 2.5.

Layers in ANN

Many neurons together constitutes a *layer*. In addition to the input and output layer, a DL model has multiple *hidden layers*, which are all the layers not directly visible from the outside, i.e. all layers except the input and output. The following paragraphs give a short explanation of the layers constituting the ANN used in this

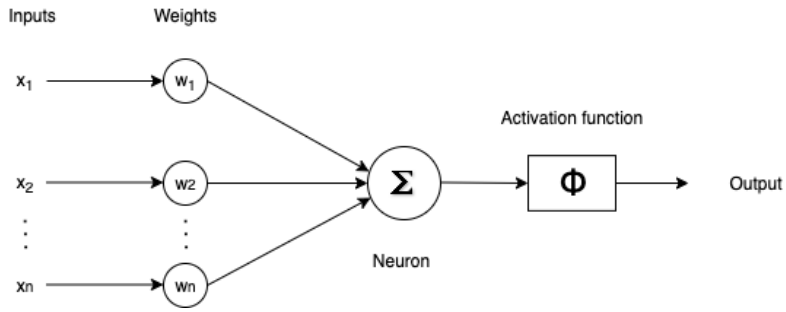

Figure 2.5: Perceptron.

thesis. Each layer extracts features from the layer below and produces an output with a higher level of abstraction to the layer above.

Dense layers: Dense or fully connected, refers to layers where all neurons are connected with every neuron in the preceding layer.

Flattened layer: One-by-n vector containing all the outputs from every node in the previous layer.

Convolutional layer: The characteristic element of a CNN is the convolution layer, which is similar to a perceptron layer, but its task is to learn features. A CNN can successfully capture spatial and temporal dependencies in the input data by the application of relevant filters. In a convolutional layer, the dot product between the input data and a *filter* is computed over a spatial region. The size of the filter should be adjusted to the structure of the input data. How the kernel shifts over the input data are set by the *stride length*. The process is continued until the entire input data is traversed.

Max-pooling layer: A max-pooling layer is used to down-sample the feature representation obtained in previous layers. It is commonly used in combination with a convolutional layer. In max-pooling, the maximum value from a spatial region or window of the input data is returned. Max pooling can help reduce computation time and costs significantly. Furthermore, it is useful for extracting dominant features.

Training an ANN

Training the model simply means learning good values for all the weights. Loss is the penalty for a bad prediction. That is, loss is a number that indicates how bad the models prediction was on one specific input. The loss function used in this work is Mean Square Error (MSE). To calculate the MSE eq. (2.8), all squared losses for

individual examples are summed up:

$$MSE = \frac{1}{N} \sum_{(x,y)\epsilon D} (y - prediction(x))^2 \tag{2.8}$$

Optimal values for the connected weights are found through backpropagation. The error for a specific input is computed, and the weights adjusted accordingly. This process is repeated for many input examples through several training iterations in the process of finding the ideal values. In this way, a ANN is self-trainable.

Autoencoders

An autoencoder is a ANN that copies its input to its output. The autoencoder learns how to compress and encode data and how to reconstruct the data back from the reduced representation. The *encoder* reduces the input dimensions to the *bottleneck* layer, which is the layer that contains the compressed representation. The *decoder* reconstructs the data from the encoded representation to be as close to the original input as possible. The *reconstruction loss* measures how close the output is to the original input, i.e. how well the model is performing. The training procedure does not require any labelling of the data; it is therefore regarded as an unsupervised learning algorithm.

The autoencoder has many applications, such as dimensional reduction, image processing and anomaly detection. The latter can be used for one-class classification. Since the model learns to precisely replicate features from one class, the reconstruction error will increase when facing data from other classes.

2.4 Optimization problems

2.4.1 Multi-objective optimization problem

A MOOP has several objective functions which are to be optimized. The problem usually has some constraints that any feasible solution must satisfy [29]. A MOOP is defined as

$$\begin{array}{lll} \text{Minimize/Maximize} & f_m(\mathbf{x}), & m = 1, 2, ..., M \\ \text{subject to} & g_j(\mathbf{x}) \leq 0, & j = 1, 2, ..., J \\ & h_k(\mathbf{x}) \leq 0, & k = 1, 2, ..., K \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, & i = 1, 2, ..., n \end{array}$$

The solution is a vector $\mathbf{x}$ with n decision variables: $\mathbf{x} = (x_1, x_2, ..., x_n)$, where all variables must take a value within the lower and upper bounds that are defined by $x_i^{(L)}$ and $x_i^{(U)}$. The functions $g_j(\mathbf{x})$ and $h_k(\mathbf{x})$ are the constraint functions that any solution must satisfy. The M objective functions that are defined in $f_m(\mathbf{x})$ can either be minimized or maximized.

2.4.2 Genetic Algorithms

In computer science a GA is a procedure used to solve optimization and search problems, inspired by the process of natural selection.

There is no strict definition of a GA, [30]. However, most methods called GA have some main elements in common: populations of *chromosomes*, *selection* according to fitness, *crossover* to produce new offspring, and random *mutation* of new offspring. The chromosomes are usually bit strings, where each locus (gene position) can be either 0 or 1. The GA evaluates the population of chromosomes and replace the populations with new ones. The fitness function gives a score to each chromosome. This score, often referred to as fitness, describes how well the chromosome solves the problem. Using this approach, the GA can replace the poorest chromosomes in order to find the optimal solution.

The selection of the chromosome for reproduction is made by the *selector* operator. The fittest chromosomes are most likely to be selected. Variation in the population is created by the *crossover* and *mutation* operator. The crossover operator mixes two chromosomes to create two new offsprings, and the mutation operator randomly flips some bits in the chromosome.

Chapter 3
State of the art

Previous research has explored EEG-based biometric systems and demonstrated that EEG-signals are unique for individuals and can be used for identification [31]. EEG is a field which has seen a lot of research over the past decade, and the use of EEG as a biometric is an emerging topic and may open for new applications in the future. This chapter aims to give an overview of the methods used in related studies. As this work is limited to regard subject authentication, the most relevant work is concerning the authentication approaches.

3.1 Pilot study

A pilot study [4] was conducted by the author in preparation for this research. The objective was to investigate methods for feature extraction and classifiers in use for a EEG-based biometric *identification* system. The tested methods were DWT and PCA. Two different datasets, recorded from ERP, with 26 and 16 subjects were used for the experiments. The study aimed to find a suitable mother wavelet and level of decomposition when using DWT for EEG signals. The result showed that both DWT and PCA were good methods as the classification accuracy was 1.0 and 0.93 when using DWT and PCA, respectively. The study addressed the need for experiments on the authentication layer for rejecting subjects that were not a part of the user-pool. Additionally, the study suggested that more thorough experimentation should be done to find the most informal channels associated with the chosen paradigm. These topics were the basis in the design of this thesis.

3.2 Paradigms

Depending on the methodology of data recording, the studies can be separated into different categories. The authors may use resting-state data [32, 33, 34], VEP [35, 36], ERP [35, 37] or imagined movement/tasks [36, 38]. In [38], the authors state that some paradigms are better for specific applications. In their study, different mental

tasks for authentication is compared, and the findings imply that some paradigms may be more suitable for authentication than others. However, this may be dependent on the datasets that are used.

3.3 Feature extraction

Feature extraction is a critical stage in the analysis of EEG-signals. The features can be classified based on domains (time, frequency and time-frequency domain) or channels (single-channel and two-channel) [10]. A variety of different methods for feature extraction has been examined in related studies.

The Autoregressive (AR) model is a widely used time-domain feature in EEG biometrics and used in a lot of studies [32, 33, 39, 40]. As described in section 2.1.4, EEG signals can be separated into frequency bands which are related to specific activities. By transforming the EEG data into the frequency domain one can extract dominant frequency components. Power Spectral Density (PSD) measures the distribution of signal strength in the frequency domain and is used as a feature in [38]. Fast Fourier Transform (FFT) is a popular method for transforming the EEG data to the frequency domain, used in [32, 33, 38]. DWT provides both time and frequency information of a the signal, the method is used for feature extraction in [33]. From the sub-bands, the authors extracts both time and frequency domain parameters. In addition, statistical parameters on the amplitudes were computed. Another method for decomposition is EMD, the method is used for extracting IMFs in [37]. From the IMFs, fractal and energy features, such as IE, TE, HFD, and PFD, is computed. Other features are also used for authentication, like skewness and kurtosis in [34], two-channel features of Mutual Information (MI) and coherence in [32]. In [33] hjort parameters (activity, mobility and complexity) are tested.

3.4 Channel reduction

A large number of channels can contain redundant and useless information. In addition to this, large datasets increase computational complexity, which can be a challenge for real-time applications. Selecting the most relevant data by using more effective channels can be a solution to this problem. Different methods for reducing the number of channels, as well as finding the most efficient channels, have been investigated in earlier studies.

A backward-elimination is presented in [41, 42, 43]. The greedy algorithm removes one channel at a time by performing the classification step and selecting the subset of channels that gives the highest accuracy. The authors of [43] also tests the opposite method, forward-addition. The algorithm creates a classifier for each channel and adds the channel with the highest accuracy to the subset. In [39], different subsets

of 3 channels are systematically tested to find the channel triplets with the best performance. It should be noted that these studies address the identification problem, not authentication.

In [37] and [44] a GA is applied to optimize channels. The result of both studies substantiates that using a smaller but more effective channel subset can improve the performance of the system. In [37] the TAR and TRR was increased from 0.92 and 0.08 to 0.95 and 0.93 using seven channels instead of 56. (It is important to mention that other parameters were optimized by the GA in this experiment as well). In [44], the classification accuracy was improved from 0.96 to 0.97 by reducing the number of channels from 64 to 37.

In [45], the authors address the problem of reducing the number of required channels while maintaining comparable performance. They evaluated a binary version of the Flower Pollination Algorithm (FPA) under different transfer functions to select the best subset of channels that maximizes the accuracy. The method makes use of less than half the number of channels (originally 64) while maintaining recognition rate up to 0.87.

The issue of real-time application and channel reduction is also issued in other work. However, the choice of channels is based on convenience and user-friendliness, not efficiency. In [33], a single-channel mobile EEG is used for data acquisition. In [32], only two frontal lobe channels are used. Those specific channels are chosen because the placement is accessible and comfortable for the user.

3.5 Authentication methodology

A lot of different methods have been explored for creating good classifiers for authentication purposes. In [32], the authors used Fisher's Discriminant Analysis (FDA) to find several projection directions that are efficient for discrimination, that is, separation in classes. When a subject is classified, they fuse the results from the five best classifiers to take the authentication decision, based on a threshold. Discriminant analyzers are used in [34] as well. In addition, they use Decision Tree (DT) and couple based classifiers. For each subject they select the classifier that preforms best. The authentication decision is taken based on a set of posteriors that is compared with a threshold. The work presented in [33] uses k-Nearest Neighbors (k-NN) classifiers to find a match between the incoming EEG signal, with one of the templates that are already stored in the database. Their work illuminates the issue of maximizing both the TAR and TRR simultaneously. The use of thresholding in combination with a CNN is tested in [35]. Their CNN has two output nodes, representing the probability for a subject being an user or an intruder. In [37], a OC SVM is used for one-class classification of users and intruder. Only one classifier is

created for the entire user-group of 13 subjects. The same approach is used in [46], where one single classifier is created for all users. However, the authorized personnel is a group of just 2-3 persons. The method used for classification is ANN. The results of the mentioned studies are presented in table 3.1.

Table 3.1: Comparison of results found in simular work.

Source	[32]	[33]	[34]	[35]	[37]	[38]	[46]
Paradigm	Resting	Resting	Resting	VEP, ERP	ERP	imagination task	imagined speech
No. subj	51 users 36 intruders	11	50 user 20 intruders	15 users 15 intruders	26	9	2 users 30 intruders
No. chans	2	1	2	16	7	32	128
Feats	AR, FT, MI, coherense and cross-correlation	DWT, statistical, FT, Hjort, AR	14 frequency- and time-domain features	ERP features, morphological features	IWE, TWE, HFD, PFD	PSD, FFT	Statistical
Clf	Fisher Discriminant Analysis	KNN	DA, DT, couplebased classifier	CNN	OCSVM	Gausian Mixture Model	ANN
Length signal	3 min train 1 min eval	1 min	2 min train few sec eval	15 min	39 sec for training	4 min	6 min
Result	TAR 0.966 FAR 0.034	TAR 0.10 TRR 0.80	TAR 0.938	TAR 0.924 TRR 0.961	TAR 0.95 TRR 0.93	TAR 0.928 TRR 0.903	Acc 0.90

3.6 Deep learning and EEG

DL has revolutionized the field of image and speech classification, but DL methods have not yet shown convincing improvement over state-of-the-art Brain Computer Interface (BCI) methods [47]. However, recent advances in the field of ANN have made them more attractive for analyzing EEG signals. The studies discussed in this section focuses on the use of DL for EEG classification.

A lot of studies show that CNN has been successfully used for EEG based classification. Some examples are motor imagery [48, 49], epileptic detection [50, 51], memorizing [52] and driver performance [53].

The performance of a CNN is closely related to its architecture design. A major challenge is to determine the appropriate depth of the network, i.e. the number of hidden layers. A lot of the researchers use 1, 2, or 3 convolution layers [49, 53, 50, 51]. Some authors make use of more layers, such as 5 in [48] and 7 in [52].

In a study regarding mental task classification [54], the authors state that convolution within a single channel is more effective than treating the signal as a whole. They argue that this method can produce information that is more valuable and free of noise from other channels and propose a multi-channel CNN design. The same design approach has been successfully explored in [55, 56]. In [56], a public available architecture, EEGNet, is presented. The compact CNN architecture preforms a convolution within one channel in the first layer. Then a depth-wise convolution for

each of the output in the first layer is preformed. This allows for extracting spatial features between channels, which means finding frequency-specific features. The EEGNet generalizes across different paradigms and for different classification tasks [56].

The state-of-the-art reports different approaches for subject authentication presented in table 3.1. The studies vary in the selection of features, classifiers, paradigms and design of the model, i.e. using one model for each enrolled subject or one common model for the entire system. Few of the works focus on real-time, and the length of EEG signal used for training is not optimal for a real-time application. In this work, a different combination of protocols, features, and classifiers are tested in the quest of finding the best combination for a real-time implementation. Also, channel selection is explored to optimize both the efficiency regarding time and classification performance.

Chapter 4
Materials and methods

The following section describes the material and methods used in this work. First, the two different datasets are described briefly. Then, the overall system layout is outlined, followed by a section where the methods for feature extraction and use of classification to create an authentication model is described in detail. Finally, a summary of the software implementation is given.

4.1 Datasets

Initially, the plan for this project was to conduct a physical experiment and record EEG-data in real-time. Due to the Covid-19 pandemic, such an experiment was impossible to perform. Thus, public available datasets have been used to test the system instead. The datasets are recorded under different paradigms, but for comparison causes, periods, where the subjects are in resting state, have been extracted from both.

The first dataset, from now on referred to as *P300*, consists of 26 subjects (13 male and 13 female, mean age = 28.8 ± 5.4, range 20 -37). EEG was recorded with 56 passive Ag/agCI EEG-sensors whose placement followed the 10-20 system described in section 2.1.3. Their signals were sampled at 600 Hz, but downsampled at 200 Hz.

The protocol followed to record the EEG-signals is called the P300-speller, where subjects performed a spelling task. The subjects went through five copy spelling sessions. Each session consisted of 60 trials, except the fifth, which consisted of 100.

The second dataset, from now on referred to as *spatial*, arises from a study decoding spatial attention from EEG with near-infrared spectroscopy prior information [57]. In the experiment, subjects attended to the left or right following instruction by visual stimuli. The experiments were conducted on eight right-handed males between 20 to 40 years of age (mean: $= 24.6 \pm 6.4$). One experiment consisted of 8 sessions,

and each session consisted of 24 trials. The signal was recorded at 256 Hz with a 64-electrode cap.

To avoid choosing methods and parameters customized for the specific datasets, different sessions are used for the prestudies (setting hyperparameters), and for running experiments on the system. Session 1 and 2 are used in the prestudies, and session 4 and 5 in the final experiments. The system should be able to recognize an user regardless of the state of the subject, i.e. different time of day, mood and condition. This is simulated by using different sessions for enrolling and login.

Table 4.1: Summary of datasets

Name	**No subj**	**Paradigm**	**Channels**	**Sessions**	**Trials**	**Instance Size**
P300	26	Resting-state	56	5	60	56 x 400
Spatial	40	Resting-state	64	8	24	64 x 512

4.2 Data pre-processing

The raw data from the EEG electrodes are structured into *instances*. For both of the datasets, there is a resting-state period between the tasks performed within each session. A time-series of 2 seconds was extracted from these periods and make up an instance. The number of channels and sampling rate gives instances of size 56×400 and 64×512 for the P300 and spatial dataset, respectively.

The only method of further preprocessing that is investigated, is standardization when using ANN. The dataset is standardized by removing the mean and scaling to unit variance. The standard score of a sample point is calculated as

$$z = \frac{x - \nu}{\sigma}, \tag{4.1}$$

where ν is the mean and σ is the standard deviation of the training samples.

4.3 System layout

The overall design of the complete system with authentication and identification is shown in fig. 4.1. EEG-signals from a public database is used for enrolling the subjects. The EEG-signal is segmented into instances; then features are extracted. The dataset is used to fit a classifier, which is stored in a database for later use. In the login phase, the model decides whether the EEG-signal belongs to a subject who has access to the system or not. It is important to emphasize that even though the data is not acquired in real-time as planned, the rest of the system (i.e. enrollment

and login phase) is operating in real-time. For the creation of the authentication model and the decision stage, two different approaches have been investigated.

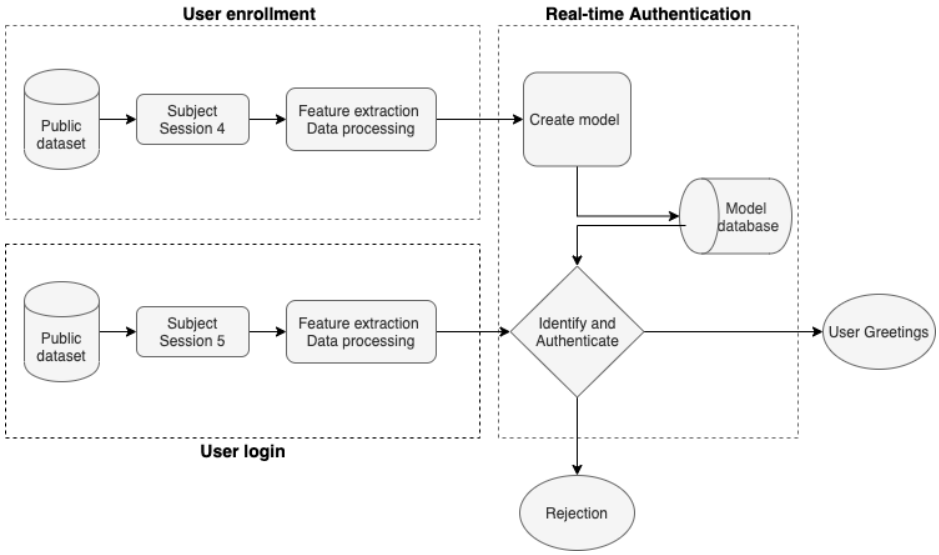

Figure 4.1: Flowchart illustrating complete system for identification and authentication during the enrollment and login phase.

Authentication methodology

Common model

The common model design proposes authentication as the first layer in the system, as illustrated in fig. 4.2. The authentication layer decides whether a subject is accepted or not. First after the subject is accepted, the system will try to predict an identity for the subject. One common authentication model is built for all the enrolled users and data from all enrolled users are used when creating the classifier. Using this design, only one classification model is required for the entire system as in [37, 46].

Subject-specific model

In the second design, one authentication model is created for each unique user. When enrolling a user, a classifier is built on training data from this subject only, see fig. 4.3. This design is proposed in a lot of similar work [32, 33, 34, 35]. When a subject attempts to access the system, the identification layer will propose an identity for the subject. The authentication model will decide whether the signal belongs to that specific subject or not. Using this design, the authentication layer is dependent on reliable prediction in the identification layer. To zero out any following error caused

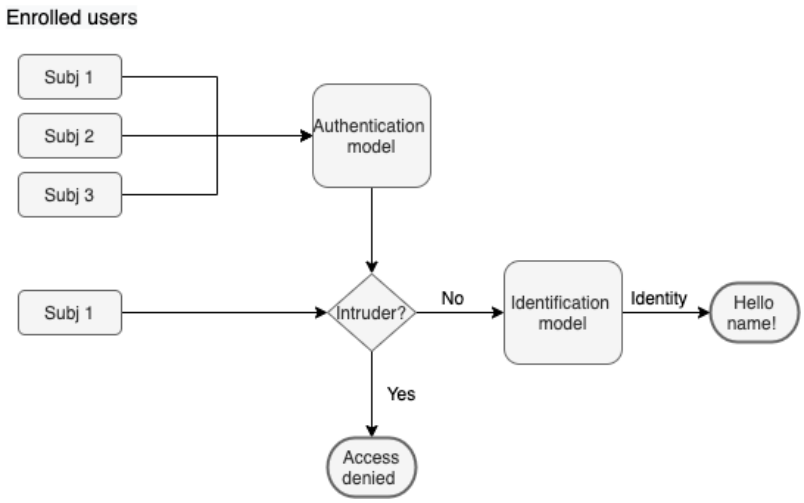

Figure 4.2: Flowchart illustrating the procedure of authentication and identification of a subject when using one common model for all users.

by wrong predictions in the identification layer, the correct id-label is always the input to the authentication layer in the experiments conducted in chapter 5. When testing for users, only the model for the specific subject can accept the user. When testing for intruders, every model must be able to reject the subject.

4.4 Feature Extraction

Three different methods for feature extraction is proposed in this work; DWT-based feature extraction, PCA and EMD-based feature extraction. A summary of all features is given in table 4.3.

DWT-based feature extraction

A flow-chart summarizing the feature extraction stage when using DWT-based features is showed in fig. 4.4. The signal from each channel is processed separately and decomposed into sub-bands. The level of decomposition is 5 for both dataset, resulting in 6 sub-bands. By decomposing the signal, we separate into frequency components, which from whom we can extract frequency-domain features. The features used are IE and TE. From each channel we extract $6 \times 2 = 12$ features. Gathering the features from each channel in a common feature vector gives 672 and 768 features for the P300 and spatial dataset when using all channels.

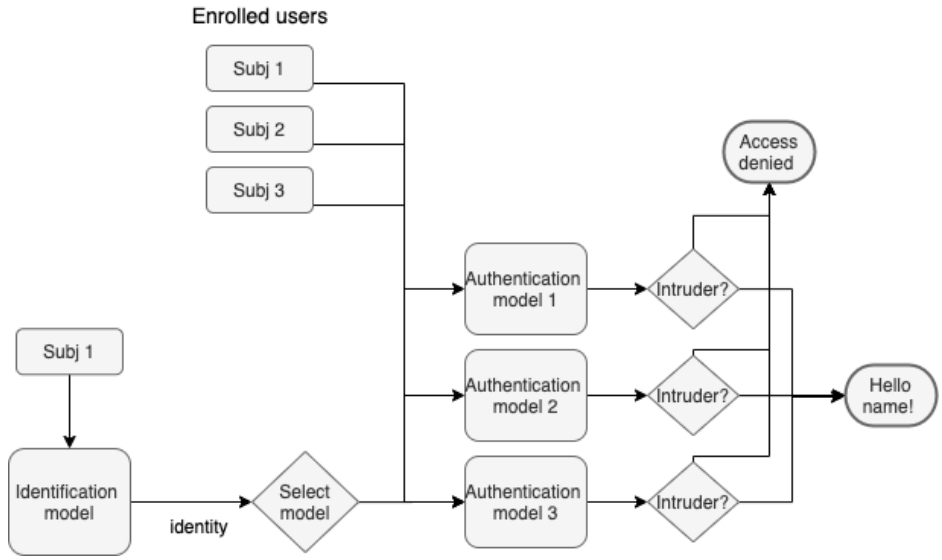

Figure 4.3: Flowchart illustrating the procedure of authentication and identification of a subject when using one model for each enrolled subject.

Decomposition

Considering that the main brain rhythms lie in the range 0.5 - 30 Hz as described in section 2.1.4 (see table 2.1), the signal is decomposed to a level of 5. The resulting sub-bands of the P300 and spatial dataset and their associated frequency ranges are presented in table 4.2. As the two datasets are sampled at different sampling rates, the sub-bands cover different frequency ranges. The mother wavelet used is *bior4.4*. The decomposition level and mother wavelet are chosen based on experiments conducted in the pilot study [4].

	Frequency range [Hz]	
Sub-band	**P300**	**Spatial**
D1	50 - 100	64 - 128
D2	25 - 50	32 - 64
D3	12.5 - 25	16 - 32
D4	6 - 12.5	8 - 16
D5	3 - 6	4 - 8
A5	0 - 3	0 - 4

Table 4.2: Frequency ranges covered by each sub-band in DWT for the P300 and spatial dataset.

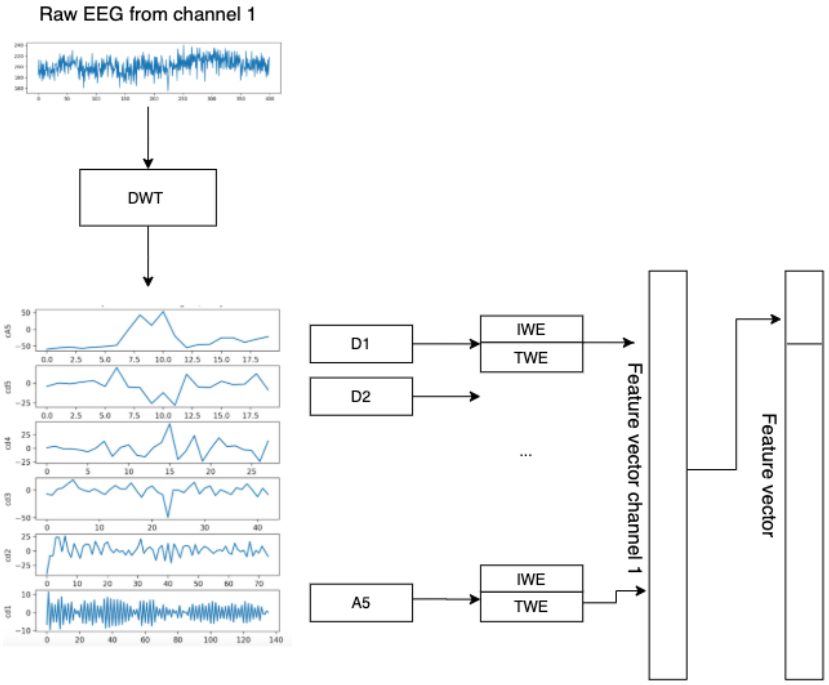

Figure 4.4: Flowchart for DWT-based features extraction, using level 5 for decomposition. This process is repeated for every channel.

Feature extraction using principal component analysis

For this method, samples from all channels are gathered in a matrix representing an individual instance. From this matrix, PCA is applied and PCs are extracted to form the feature vector.

The number of PCs used in the feature vector is based on the cumulative variance, see plots in fig. 4.5. The plots show the fraction of variance explained by each of the PCs. A threshold of 95% is marked in the plots to show how many PCs that should be included to preserve 95% of the total variance in the data. As a PCA is done on each instance individually, how many components that are needed varies for each instance. All instances must have equal-sized feature vectors, meaning we must extract an equal amount of PCs. Considering efficiency and performance, the smallest number of PCs possible should be chosen. From the plots, one can see that by selecting two components in the P300 dataset, around 95% of the variance will be retained for most instances. For the spatial dataset, one PC is enough. However, two PCs have been used for both datasets in the experiments for simplicity. Using 56 channels and two principal components, the length of the feature vector can be

calculated as $56 \times 2 = 112$.

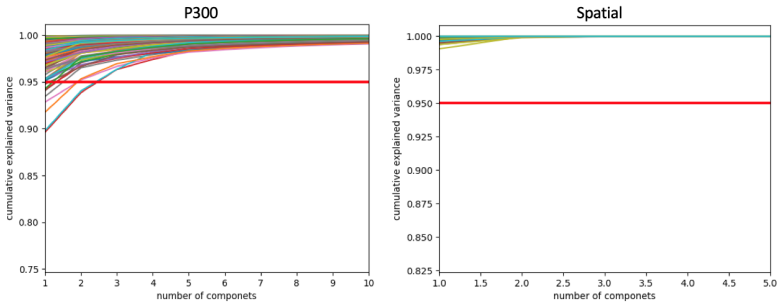

Figure 4.5: Cumulative explained variance for all instances in both datasets.

Feature extraction using Empirical Mode Decomposition

When using EMD for feature extraction, the signal from each channel is decomposed into IMFs using the EMD algorithm described in section 2.2.3. The number of IMFs that can be extracted may vary in different channels and instances. As all feature vectors must be of the same size, a finite set of IMFs is chosen. For this thesis, the first two are selected. From both IMFs, four energy and fractal features are calculated. This means that for every channel, a set of $2 \times 4 = 8$ features are extracted, resulting in a vector size of 448 for the P300 dataset and 512 for the spatial using all channels.

Table 4.3: Summary of the feature extraction methods. The vector sizes are for using all channels.

Method	**Features**	**Vector size**	
		P300	**Spatial**
DWT	IWE, TWE	672	768
PCA	PC	112	128
EMD	IWE, TWE, PFD, HFD	448	512

4.5 Classification

The authentication of a subject is solved as a one-class classification problem. Two different ML models are used for classification; the first method uses the features extracted in the previous section, while the other uses DL and learns features directly from the raw data.

Classification using OC SVM

Once the feature vector for each instance is extracted, the process is similar for the three feature extraction methods. The feature vector is fed to a OC SVM, which trains on the unlabeled data. When using the common model layout the OC SVM trains on data from all enrolled subjects and when using the subject-specific model only data from one subject is used. The hyper-parameters *nu* and *gamma* are preset and selected from an optimization problem described in detail in section 4.6. The overview of the process is illustrated in fig. 4.6.

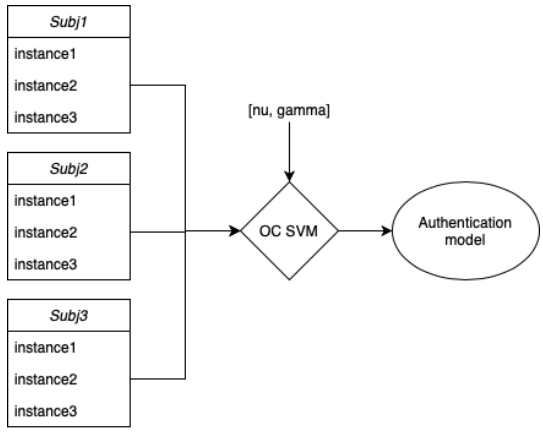

Figure 4.6: Authentication model using OC SVM and common model layout.

Classification using deep learning

The second approach for the authentication model uses a threshold value to determine if a subject is an user or not. When enrolling a subject an autoencoder is created. The autoencoder learns how to compress the data from the specific user and reconstruct it. A 50/50 split of the recorded EEG is used when enrolling a subject, in which 50% of the data is used to train the autoencoder. The remaining 50 % is passed through the autoencoder to find a suitable threshold for the reconstruction error. In the login phase, the error between the original instances and the reconstructed ones are compared against this threshold value. The autoencoder and the threshold for the reconstruction error constitute the authentication model, see fig. 4.7.

In fig. 4.8, the reconstruction error for a set of instances are plotted. Data from the user class is in green and the remaining in red. As the plot shows the reconstruction error for the user class is much smaller than for the rest of the subjects, which indicates that a good threshold value can be found. To choose a threshold for the error, Part Average Limit (PAL) [58] was used as a basis. To find the PAL, the reconstruction error for each of the instances used in the test set under login is

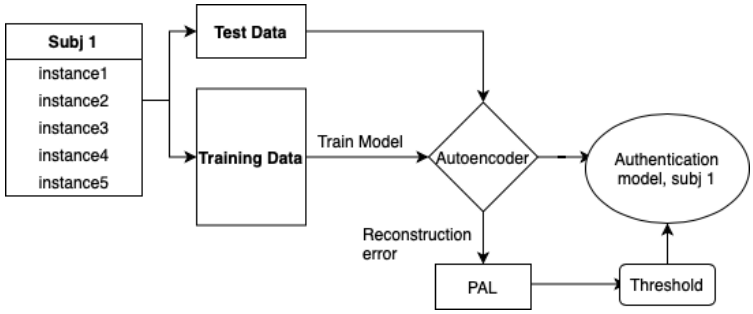

Figure 4.7: Authentication model using an autoencoder and threshold. The layout is for the subject-specific model.

calculated. The median and Interquartile range (IQR) of this distribution is used to find a value that allows for variation. The IQR is a measure of statistical distributions, and is equal to the difference between the 75th and 25th percentiles [59]. The PAL is calculated as in eq. (4.2).

$$PAL = median \pm C \times \frac{IQR}{1.35} \tag{4.2}$$

The value of C is determined experimentally, the values can be seen in chapter 5. As it is only meaningful to talk about positive values for the error, only the upper PAL is used.

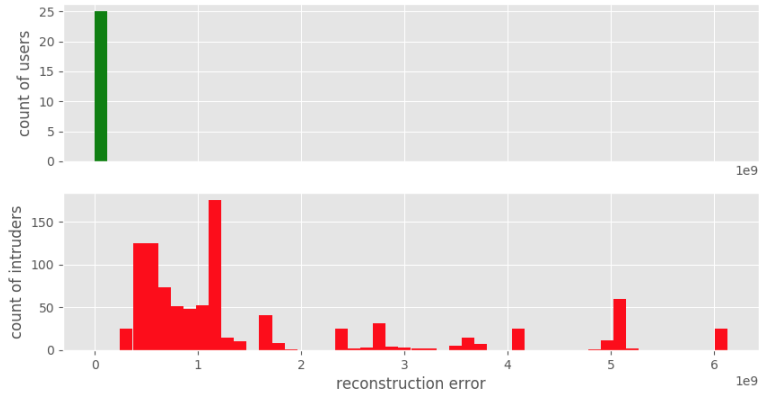

Figure 4.8: Reconstruction error for autoencoder. The model is trained on subject 2. User data plotted in green and intruder data in red.

Architecture for the autoencoder

The architecture used in the encoders of this work is inspired by CNN used in similar work, presented in section 3.6. The decoder was built with the inverse operations to reconstruct the compressed data. Two different architectures for the autoencoder has been investigated. Both models are fitted using the Adam optimizer, minimizing the MSE of the difference between input and output data.

Autoencoder with CNN

The first neural network gathers data from all channels into an EEG data matrix of size $channels \times instance\_length$. The signal has one feature for each sampling; the EEG voltage. Hence, the size of the input vector is $56 \times 400 \times 1$ for the P300 dataset and $64 \times 512 \times 1$ for the spatial. The encoder performs two 2D convolutions on the matrix with filters of different sizes to extract spatial features. Each convolution layer is followed by an activation layer, LeakyReLu. The encoder is illustrated in fig. 4.9. The decoder performs the same operations, only inverse. A detailed description of each layer in the autoencoder is summarized in table 4.4. The output of the autoencoder is a reconstructed matrix of the same shape as the input.

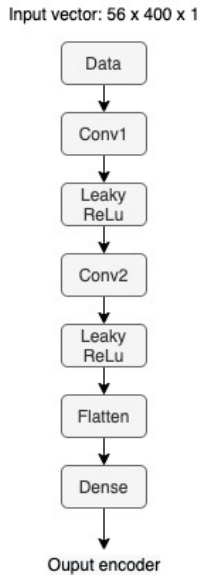

Figure 4.9: Layers in the encoder of the CNN autoencoder. The size of the input vector is when using P300 data.

Multi-channel autoencoder

The second autoencoder is built around the idea presented in [60], that the convolution of a single channel can produce information that is more valuable and free of noise

	Layer	Size of output	No filters	Filter size	Strides
Encoder	Input	(56, 400, 1)			
	Conv2D	(28, 200, 32)	32	(3, 3)	2
	LeakyReLu	(28, 200, 32)			
	Conv2D	(14, 100, 64)	64	(3, 3)	2
	LeakyReLu	(14, 100, 64)			
	Flatten	(89600, 1)			
	Dense	(16, 1)			
Decoder	Input	(16, 1)			
	Dense	(89600, 1)			
	Reshape	(14, 100, 64)			
	Conv2d Transpose	(28, 200, 64)	64	(3, 3)	2
	LeakyReLu	(28, 200, 64)			
	Conv2d Transpose	(56, 400, 32)	32	(3, 3)	2
	Activation (sigmoid)	(56, 400, 1)			

Table 4.4: Layer summary for CNN autoencoder when using P300 data.

from other channels than if the signal is treated as a whole. The autoencoder separates the signals and performs 1D convolution on the isolated channels. The design prevents information mixing between the individual channels. An illustration of the data flow in the encoder is visualized in fig. 4.10. In each branch of the encoder, convolution is performed, followed by pooling for downsampling and finally, an activation layer. The signal is flattened before all branches are concatenated. The inverse operations are performed in the decoder to reconstruct the signals. All layers are described in table 4.5.

4.6 Optimization problem for finding best hyper-parameters and channels

As described in chapter 2, ML algorithms have internal settings called hyper-parameters. By finding suitable hyper-parameters, one can maximize the learning. In order to find the best nu and gamma for the decision boundary of the OC SVM, an MOOP was designed. This method allows for adjusting the parameters according to results after testing on both user and intruder data. In this way, one can find optimal parameters for good rejection and acceptance performance. The goal is to find a decision boundary that will enclose all users and leave all intruders outside.

The performance of the systems is measured by the TAR and TRR. Hence, the task is to find what nu and gamma that maximizes both of these values. Also, as suggested in section 3.4, some channels may be more useful for specific classification tasks. If the removal of a channel increases the performance, this channel should be

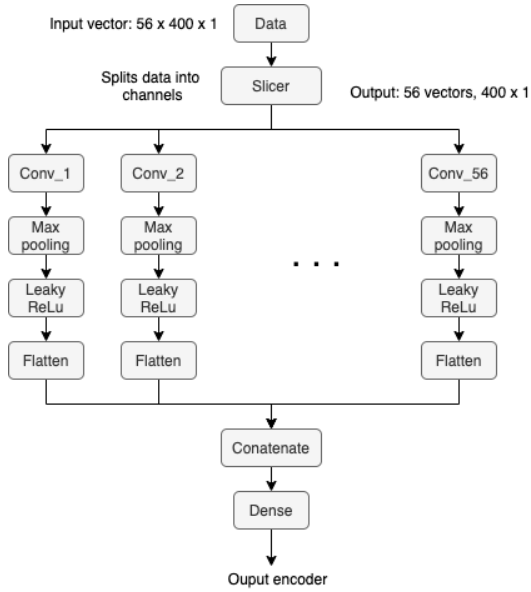

Figure 4.10: Layers in the encoder of the multi-channel autoencoder, treating signal from each channel separately in branches. Vector size is when using P300 data.

excluded. Accordingly, reducing the number of channels was added to the problem. The three objectives are written in eq. (4.3).

$$\begin{aligned} max f_1(x) &= TAR \\ max f_2(x) &= TRR \\ min f_3(x) &= num\_channels \end{aligned} \tag{4.3}$$

The decision variables are *nu*, *gamma* and a channel array of size $num\_channels$. As described in detail in section 2.3.2, nu and gamma denotes the strictness of the classifier and are real values between 0 and 1, thus, a lower bound of 0.00001 and an upper bound of 0.9 is set to limit x_1 and x_2.

The remaining decision variables in the channel array are binary, where 1 indicates that the channel with this index should be included in the channel array and 0 indicates excluded. This results in an solution vector $\mathbf{x}$ of size $2 + num\_channels$.

	Layer	Size of output	No filters	Filter size	Strides
Encoder	Input	(400, 1)			
	Conv1D	(400, 32)	32	(10, 1)	2
	Max Pooling	(200, 32)		(2, 1)	
	Leaky ReLu	(400, 32)			
	Flatten	(6400, 1)			
	Concatenate	(358400, 1)			
	Dense	(16, 1)			
Decoder	Input	(16, 1)			
	Dense	(358000, 1)			
	Split	(6400, 1)			
	Reshape	(200, 32)			
	Up sampling	(400, 32)		(2, 1)	
	Conv1D transpose	(400, 1)	32	(10, 1)	1
	Activation (sigmoid)	(400, 1)			

Table 4.5: Layer summary for each branch in the multi-channel autoencoder using P300 dataset.

Minimum one channel must always be chosen ($num\_channels \geq 1$); therefore, an inequality constraint is defined for the problem. Rewriting to a less-than-constraint, $g(x)$ can be written as in eq. (4.4).

$$g(x) = 1 - num\_channels \leq 0. \tag{4.4}$$

The problem is solved using the GA NSGA2. The population size is 100 for each iteration, and the experiment is repeated for 50 generations. The chromosome is of the same size as the solution vector, with real values for nu and gamma and binary values for the channel array. An example chromosome for 10 channels is illustrated in fig. 4.11. The samplings are created randomly both for the real and binary genes. Binary crossover is used for both real and binary values, and the mutation method used is bit flip.

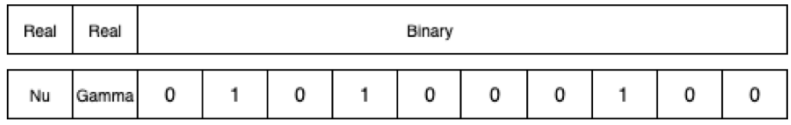

Figure 4.11: Example of chromosome in the population when using 10 channels. In this example, channels with index 1, 3 and 7 should be included.

4.7 Application implementation

The following section gives an outline of the design and implementation of the python application used to realize the EEG biometric system.

The system consists of a client/server RESTful API. A full overview of the endpoints for the communication is given in appendix A. The server is implemented in collaboration with another student [61]. The client and interface were fully implemented in that work and will not be described further in this thesis.

Server

Django is a Python web framework that was chosen for the server. Django is built on the MVC pattern and splits its modules into applications. Five small applications were implemented for this system:

- **common _app** - creates all the models for the database.
- **subject** - handles subjects administration, such as adding and removing users.
- **data** - handles all data processing, such as extracting features.
- **identification** - identify the subject.
- **authentication** - decide whether to grant access to a subject or not.

Database

Django comes with built-in Object-Relational Mapping (ORM). This allows for easily executing database operations in Python. A MySQL database was designed and implemented for the project. The entity-relationship diagram in fig. 4.12 gives an overview of the database architecture.

The `application` entity is the basis for the system. It allows for creating multiple separate login systems, which means that one application can be created for the two different datasets. In a bigger context, applications can be created for different use, such as one application for NTNU and another for another organization. The application is stored with a name and additional info describing the system. Also, the number of channels used for recording the EEG is saved, as this may vary. In an application there exists multiple `subjects`, `catalogs` and `authCatalogs`. Hence these entities are linked to the application using a foreign key.

The `subjects` entity stores all the users of the system. Which application the subject belongs to, is given by the foreign key. Additionally, the name of the subject is stored to greet the user when he or she accesses the system.

When a subject is enrolled the instances is stored as records in the `instances` table. The instances are linked to the specific subject via the `subjectId`. As the autoencoders uses the raw data and the OC SVM extract features, both raw data and features are saved. The method used for feature extraction (DWT, PCA or EMD) is stored in the `feature type` attribute.

In the `AuthCatalog` table, the authentication models are stored. The `AuthCatalogId` and `ApplicationId` is the primary and foreign key. The actual classifier is saved in the `model` field. What kind of features extraction method that is used is given by the `feature type`. What subject the authentication model is trained to recognise is given by the `subjectId` attribute. Depending on the authentication methodology, some of the fields are left empty. For instance, when using the autoencoder, the field for `feature type` is empty as raw data is used directly. The same applies to the `subjectId` when the common model structure is used. In that case, the `authCatalog` is applicable to the entire system and no `subjectId` is necessary.

The `catalog` table stores the model used for identification and is not relevant for this work.

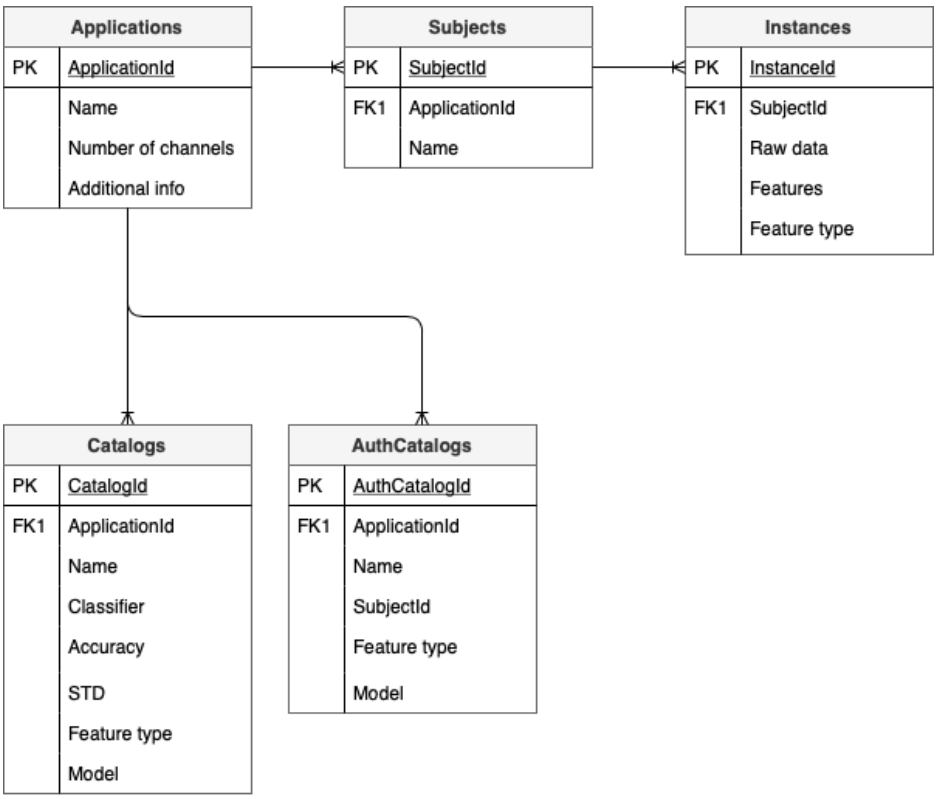

Figure 4.12: Entity relationship diagram for the complete biometric system.

Chapter 5
Results and discussion

Several experiments have been designed to test the potential of the methods outlined in the previous chapter. The methods are tested on both datasets presented. There are two terms that are essential to evaluate the system: *User attempts*, the login situation where the subject is already enrolled in the system, and *intruder attempts*, where an unknown subject tries to log in. The performance of the system is measured in the ability to accept enrolled users and reject intruders, given by the TAR and TRR. Throughout the chapter, the results are discussed.

5.1 Finding hyperparameters for OC SVM

In the search for good hyper-parameters, the optimization problem described in the previous chapter is solved. A GA selects values for nu and gamma, in addition to a channel subset to maximize the TAR and TRR. The experiment is conducted for both the common model and the subject-specific model. In the experiments, the models are trained in 24 instances. In the login phase, only one instance is used. The computations were performed on resources provided by the NTNU IDUN/EPIC computing cluster [62].

Common model

For experiments with the common model layout the subjects are split into two equally sized subsets; intruders and users, which results in groups of 13 for the P300 dataset and 20 for the spatial dataset. 10-fold cross-validation is used for testing different subdivisions of users and intruders to avoid any bias to the selected user set. After training the model, each subject in the set attempts to access the system 24 times, thus $24 \times n_u$ user attempts are performed and $24 \times n_i$ intruder attempts. The general nu, gamma, TAR and TRR is computed as the average from all folds. The results are presented in table 5.1.

Dataset	Method	Nu	Gamma	TAR	TRR	Chans
P300	DWT	0.099 ± 0.17	0.176 ± 0.18	0.522 ± 0.42	0.591 ± 0.44	4 ± 6
	PCA	0.231 ± 0.21	0.059 ± 0.13	0.127 ± 0.24	0.903 ± 0.23	1 ± 2
	EMD	0.170 ± 0.25	0.329 ± 0.31	0.658 ± 0.35	0.571 ± 0.32	10 ± 2
Spatial	DWT	0.076 ± 0.10	0.267 ± 0.18	0.342 ± 0.31	0.869 ± 0.29	5 ± 6
	PCA	0.534 ± 0.14	0.121 ± 0.19	0.321 ± 0.21	0.864 ± 0.23	2 ± 4
	EMD	0.154 ± 0.23	0.287 ± 0.28	0.514 ± 0.36	0.598 ± 0.36	9 ± 2

Table 5.1: Results for nu, gamma and number of channels when solving optimisation problem using the common model.

Even though the TRR seem to be relatively high for all the different feature methods, the model fails to obtain a high TAR at the same time. The standard deviation is high for the TAR and TRR, meaning that the results are better for some sub-divisions than others. However, a tendency is that if the TAR is high, the TRR is low and vice versa. Despite that good results are obtained in some folds, the average TAR and TRR does not meet the requirement for precision in a biometric system.

Subject-specific model

For the subject-specific model, the optimization problem is solved for one subject at a time. The experiment is repeated for every subject in the dataset, i.e., 26 times for the P300 and 40 times for the spatial. 10 random intruders are selected from the remaining subject set. The model is evaluated with 24 user attempts, and 24×10 intruder attempts. In the final system, individual values can not be used for each user. Hence, the nu and gamma found for each experiment are averaged over all subjects. Again the general TAR and TRR is averaged from the result of each experiment. All results are presented in table 5.2.

Dataset	Method	Nu	Gamma	TAR	TRR	Chans
P300	dwt	0.140 ± 0.15	0.139 ± 0.20	0.737 ± 0.26	0.938 ± 0.12	2 ± 8
	pca	0.199 ± 0.24	0.722 ± 0.33	0.678 ± 0.02	0.98 ± 0.03	1 ± 1
	emd	0.174 ± 0.21	0.203 ± 0.27	0.798 ± 0.31	0.931 ± 0.11	2 ± 5
Spatial	dwt	0.063 ± 0.05	0.061 ± 0.12	0.969 ± 0.06	0.979 ± 0.06	1 ± 3
	pca	0.158 ± 0.17	0.133 ± 0.23	0.726 ± 0.21	0.923 ± 0.13	1 ± 3
	emd	0.091 ± 0.12	0.192 ± 0.21	0.897 ± 0.19	0.964 ± 0.07	1 ± 9

Table 5.2: Results for nu and gamma values found solving optimization problem using the subject-specific model.

The results when using the subject-specific model is significantly improved compared to the common model for all methods. The standard deviation in the TAR and TRR is lower than for the common model, which means the results seem to be more stable

and independent of the division into users and intruders. The results presented in table 5.2 show that high precision can be obtained using a few channels: TAR and TRR of 0.969 and 0.979 for 40 subjects is obtained using spatial data and DWT-based features, this when 1 ± 3 channels are used.

5.2 Channel selection

The distribution of the channel selection from the previous section is studied to find optimal channels regardless of the user-intruder division of the subject set. The plots in fig. 5.1 and fig. 5.2 shows how many times each channel is picked as an optimal channel. An experiment to see if using these channels can improve the performance is conducted. In the first experiment, all channels are used. Then, a condition is set for including the channel. The condition says how many times the specific channel should be selected as a part of the optimal subset. The choice of condition is based on the channel scores presented in fig. 5.1 and fig. 5.2.

The nu and gamma values found for each feature extraction method in the previous section are used. The channels, as well as the nu and gamma values, were chosen when using other sessions of the dataset. Thus, the experiment will disclose how well the previous session's optimal values translate to new data.

In table 5.3, the selected channels are grouped based on their placement on the scalp. As described in detail in section 2.1.3, the channels are named after which lobe they are placed on. From the table data, one can see that channels placed at the frontal, the parietal and the central lobe is selected the most in both datasets. For the P300 dataset, the area between the central and parietal lobe is selected often. In the spatial dataset, the area between the parietal and the occipital lobe is frequently chosen.

Common model

Once again, the subject set is divided into two random sets; users and intruders. 20 subjects are used for both datasets, resulting in 10 users and 10 intruders. For training the model, 24 instances are used. Each subject in the set attempts to access the system 10 times, thus $10 \times n_u$ user attempts are preformed and $10 \times n_i$ intruder attempts. 10-fold cross-validation is used for overall results. The results can be viewed in table 5.4.

Subject-specific model

Also, for the subject-specific model, a subset of 20 subjects is used, where half of the subjects are users, and the other half is intruders. The model is trained on 24 instances. The model is tested with 10 user attempts and 10 attempts from each of

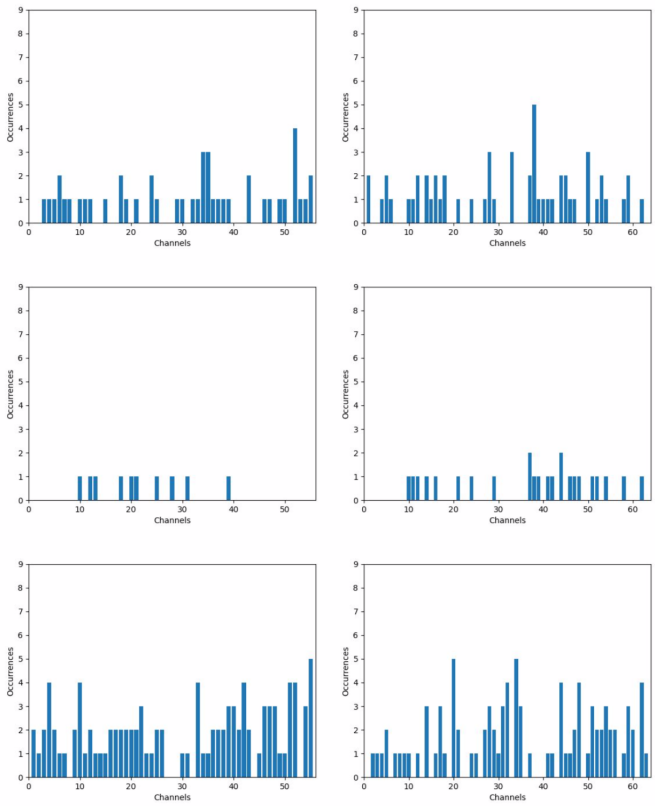

Figure 5.1: Occurrences of a channel being picked as one of the optimal channels using the common model. The first column is using the P300 dataset and the second is from spatial data. From top to bottom the features used are DWT, PCA and EMD.

the 10 intruders for evaluation. The results when using different subsets of channels can be viewed in table 5.5.

5.3 Discussion - optimal values

In this experiment, the values found in the previous section are tested on another part of the dataset. Also, aggregated values for nu, gamma, and channels are used. Hence the values are not customized for the specific user-intruder division of the subjects.

The results show that for both model types, the TAR is 0 for all methods when using all channels. When only channels that are picked by the GA is used, the

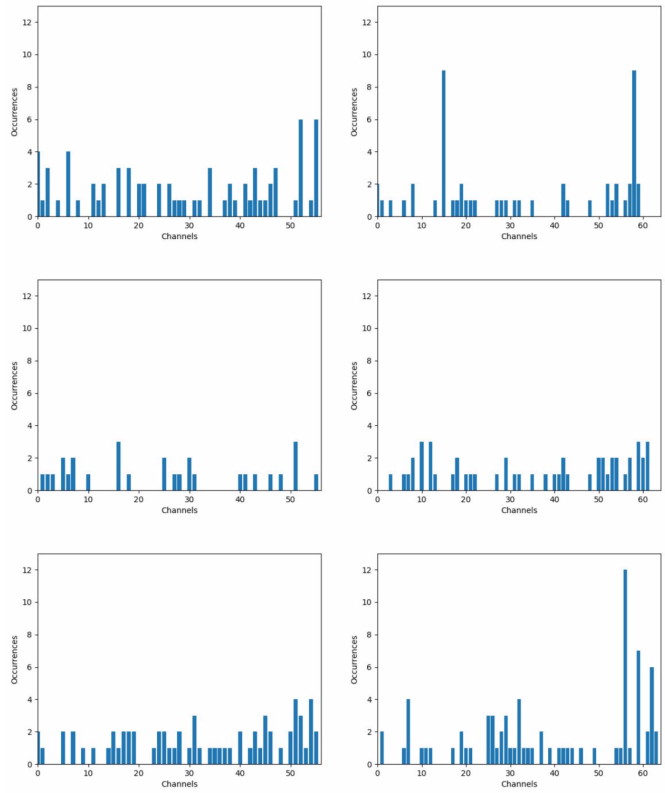

Figure 5.2: Occurrences of a channel being picked as one of the optimal channels using the single model. The first column is using the P300 dataset and the second is from spatial data. From top to bottom the features used are DWT, PCA and EMD.

TAR increases. This is common for all methods, which indicates that the result is improved by using selected channels. However, this seems to be at the expense of the rejection performance as the TRR decreases. Although this is not always the case. When using DWT in the spatial dataset a TAR and TRR of 0.64 and 0.98 is achieved when using only 2 channels. Also, when using EMD-based features, using only 2 channels gives a TAR and TRR of 0.74 and 0.92, respectively.

Compared to the results presented in the previous chapter, the performance is weaker. This might indicate that the values are customized and can not translate as well as hoped for another part of the dataset. The weaker performance may also be due to poor generalization for the nu, gamma, and channels values. It seems that the values should be chosen for each user-intruder division, and it turns out to be challenging to find universal values. However, these values should be pre-defined in the system. A solution may be to run the optimization problem in the systems operating time.

Name	Placement of channel	Occurrences	
		P300	Spatial
F	Frontal	29	27
FP	Between frontal and parietal	6	5
P	Parietal	29	29
T	Temporal	7	3
O	Occipital	9	10
C	Central	26	27
AF	Frontal	13	14
FC	Between frontal and central	23	21
FT	Between frontal and temporal	5	7
CP	Between central and parietal	25	23
TP	Between temporal and parietal	5	7
PO	Between parietal and occipital	10	28

Table 5.3: Scalp placement of the selected channels in both dataset.

When adding a new user, one can search for the best values for this specific subject. However, this is extremely computation heavy and time-consuming and is therefore not suited for a real-time application. An approach may be to use a set of default good values in the beginning, and then start a process in a different thread for the optimization problem.

5.4 Choosing threshold values for the autoencoders

As described in section 4.5, PAL (see eq. (4.2)) was used as a basis for finding a threshold value for the reconstruction error when using autoencoders. If the threshold value for the reconstruction error is too low, the model will accept intruders, and if the threshold value is too high, users may be rejected. To find a good threshold value, the coefficients were adjusted according to the obtained TAR and TRR. An example of how the TAR and TRR changes for different coefficient values is presented in fig. 5.3. The values are from an experiment using CNN autoencoder and spatial data. A customized coefficient value was chosen for each of the autoencoder models for each of the datasets and model type, the values can be viewed in table 5.6.

Dataset	Features	Nu	Gamma	Cond.	No. chans	TAR	TRR
P300	dwt	0.099	0.176	-	56	0.0	1.0
				>2	3	0.66	0.29
				>3	1	0.804	0.06
	pca	0.231	0.059	-	56	0.0	1.0
				>0	6	0.14	0.89
				>1	2	0.27	0.78
	emd	0.170	0.329	-	56	0.026	1.0
				>2	14	0.369	0.77
				>4	1	0.62	0.46
spatial	dwt	0.076	0.267	-	64	0.0	1.0
				>2	4	0.243	0.80
				>3	1	0.756	0.19
	pca	0.534	0.121	-	64	0.0	1.0
				>0	22	0.23	0.99
				>1	2	0.34	0.79
	emd	0.154	0.287	-	64	0.0	1.0
				>2	14	0.218	0.8
				>3	6	0.527	0.43

Table 5.4: Results for the common model when using a smaller subset of channels that were found by the GA. The condition column refers to how many times the channel is selected.

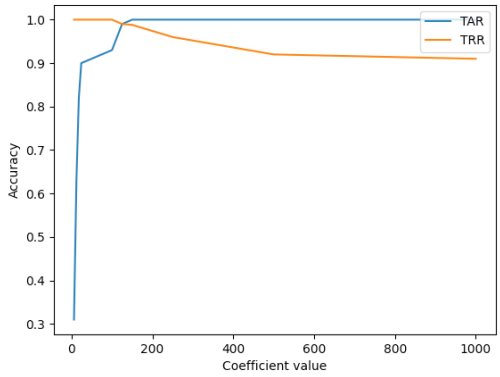

Figure 5.3: TAR and TRR when adjusting the coefficients value.

Dataset	Feats	Nu	Gamma	Cond	No. chans	TAR	TRR
P300	dwt	0.140	0.139	-	56	0.0	1.0
				>3	4	0.12	0.99
				>5	2	0.23	0.98
	pca	0.199	0.722	-	56	0.0	1.0
				>0	21	0.08	0.99
				>1	2	0.18	0.97
	emd	0.174	0.203	-	56	0.0	0.99
				>1	20	0.02	0.98
				>3	2	0.21	0.87
spatial	dwt	0.063	0.061	-	64	0.01	1.0
				>1	10	0.24	1.0
				>8	**2**	**0.64**	**0.98**
	pca	0.534	0.121	-	64	0.0	1.0
				>0	33	0.17	0.94
				>1	14	0.26	0.91
	emd	0.091	0.192	-	64	0.0	1.0
				>3	5	0.56	0.98
				>6	**2**	**0.74**	**0.92**

Table 5.5: Results for the subject-specific model when using a smaller subset of channels that were found by the GA. The condition column refers to how many times the channel is selected.

Autoencoder	Dataset	Model type	C
CNN	P300	common	6
		single	40
	Spatial	common	6
		single	125
Multi-channel	P300	common	5
		single	60
	Spatial	common	6
		single	125

Table 5.6: Coefficient values for threshold in autoencoders

5.5 Complete system test

In this experiment, the performance of all methods is compared; the results are presented in table 5.7, table 5.8, table 5.9 and table 5.10. The objective of this

experiment is to evaluate both the accuracy and the time-sensitivity for each method. In an effort to improve the real-time aspects of the system, the instances used for training are tried reduced: 24, 18, 12, and 6 instances are used for training the model. The training time presented in the tables are time elapsed in the server.

In an effort to reduce the number of channels used by the autoencoders, the channels subset found by the GA for DWT in section 5.2 is used in the autoencoders as well. Even though the channel subset is not explicitly selected for the autoencoders, the result may indicate whether the performance can be maintained when reducing the number of channels or not.

Common model

In this experiment, 10 subjects are considered as users, and the other 10 subjects are treated as intruders. 10 attempts are made for each of the subjects, resulting in 100 user tests and 100 intruder tests. The experiment is repeated for 10 random subdivisions of users and intruders. The results for all methods and P300 data can be viewed in table 5.7 and for the spatial data in table 5.8.

The multi-channel autoencoder performs best when using 24 and 18 instances. When reducing the training instances to 12 and 6, the CNN autoencoder yields the best results for the P300 data. In the spatial dataset, the multi-channel autoencoder obtains the best result in all cases. However, the training time for the autoencoders when using all channels is several minutes, which is not desirable in a real-time application. By reducing the number of channels, the training time is reduced to some seconds. Still, using four channels in the multi-channel autoencoder, requires 45 seconds for training, which is quite long.

CLF	chans	24			18			12			6		
		TAR	TRR	time	TAR	TRR	time	TAR	TRR	time	TAR	TRR	time
OCSVM dwt	3	0.65	0.42	00:01	0.59	0.53	00:01	0.50	0.50	00:01	0.37	0.72	00:01
OCSVM pca	2	0.27	0.78	00:01	0.28	0.76	00:01	0.21	0.80	00:01	0.21	0.92	00:01
OCSVM emd	14	0.29	0.80	00:01	0.29	0.65	00:01	0.26	0.68	00:01	0.28	0.87	00:01
autoencoder	56	0.96	0.18	05:05	0.92	0.12	04:49	0.82	0.04	04:42	0.89	0.0	04:39
	3	0.86	0.21	00:29	0.87	0.18	00:13	0.76	0.23	00:09	0.67	0.12	00:07
multi-channel	56	0.65	0.41	06:31	0.78	0.32	06:02	0.78	0.12	05:34	0.68	0.03	05:02
	3	0.76	0.42	00:41	0.74	0.32	00:28	0.69	0.18	00:19	0.59	0.12	00:10

Table 5.7: Results for reducing number of training instances for the common model and P300 data.

Subject-specific model

When using the subject-specific model, 10 users are selected from the dataset. The model is evaluated with attempts from the user itself and 10 attempts from 10

CLF	chans	24			18			12			6		
		TAR	TRR	time	TAR	TRR	time	TAR	TRR	time	TAR	TRR	time
OCSVM dwt	4	0.25	0.88	00:01	0.17	0.85	00:01	0.14	0.90	00:01	0.04	0.98	00:01
OCSVM pca	22	0.23	0.99	00:01	0.18	0.92	00:01	0.12	0.95	00:01	0.02	1.0	00:01
OCSVM emd	14	0.28	0.71	00:01	0.20	0.90	00:01	0.13	0.86	00:01	0.06	0.99	00:01
autoencoder	64	0.61	0.20	05:35	0.58	0.32	05:12	0.50	0.31	05:09	0.48	0.29	04:58
	4	0.84	0.18	00:32	0.78	0.12	00:14	0.59	0.08	00:10	00.39	0.07	00:07
multi-channel	64	0.70	0.60	08:32	0.82	0.54	08:23	0.65	0.45	08:01	0.53	0.21	07:51
	4	0.82	0.38	00:45	0.84	0.32	00:27	0.83	0.16	00:21	0.92	0.25	00:13

Table 5.8: Results for reducing number of training instances for the common model and spatial data.

different intruders. The results for using P300 and spatial data are presented in table 5.9 and table 5.10.

The multi-channel encoder is the method that performs best for both datasets. The training time when using all channels is shorter than when using the common model. When using only 2 channels, the training time is just a few seconds for the autoencoders, which is acceptable. The best result is obtained using the spatial dataset and only 18 training instances. This experiment yields both a TAR and TRR of 1.0 for experiments with 20 subjects.

CLF	chans	24			18			12			6		
		TAR	TRR	Time	TAR	TRR	Time	TAR	TRR	Time	TAR	TRR	Time
OCSVM dwt	2	0.23	0.98	00:01	0.22	0.98	00:01	0.12	0.99	00:01	0.04	1.0	00:01
OCSVM pca	2	0.18	0.97	00:01	0.17	0.96	00:01	0.09	0.98	00:01	0.0	1.0	00:01
OCSVM emd	5	0.14	0.94	00:01	0.135	0.94	00:01	0.12	0.94	00:01	0.0	0.98	00:01
autoencoder	56	0.87	0.57	00:32	0.71	0.92	00:21	0.66	0.93	00:15	0.56	0.89	00:09
	2	0.88	0.53	00:02	0.69	0.78	00:02	0.69	0.69	00:01	0.54	0.44	00:01
multi-channel	56	0.92	1.0	01:36	0.89	0.97	01:15	0.87	0.91	00:54	0.83	0.81	00:51
	2	0.94	0.92	00:03	0.87	0.92	00:02	0.88	0.89	00:02	0.78	0.81	00:01

Table 5.9: Results for reducing number of training instances for the subject-specific model and P300 data.

CLF	chans	24			18			12			6		
		TAR	TRR	Time	TAR	TRR	Time	TAR	TRR	Time	TAR	TRR	Time
OCSVM dwt	2	0.64	0.99	00:01	0.62	1.0	00:01	0.56	1.0	00:01	0.36	1.0	00:01
OCSVM pca	14	0.26	0.91	00:01	0.22	1.0	00:01	0.18	1.0	00:01	0.12	1.0	00:01
OCSVM emd	2	0.74	0.92	00:01	0.70	0.98	00:01	0.58	0.96	00:01	0.38	0.98	00:01
autoencoder	64	0.99	0.99	00:37	0.94	0.99	00:28	0.88	0.98	00:22	0.82	0.93	00:16
	4	0.85	0.97	00:05	0.80	0.98	00:04	0.78	0.94	00:03	0.33	0.95	00:02
multi-channel	64	0.97	1.0	01:42	1.0	1.0	01:21	0.90	0.9	01:06	0.83	0.81	00:58
	2	0.97	0.95	00:04	0.91	0.95	00:03	0.83	0.96	00:02	0.36	0.98	00:02

Table 5.10: Results for reducing number of training instances for the subject-specific model and spatial data.

5.5.1 User capacity test

To see if the performance decreases when using more subjects, a user capacity test has been designed. All subjects in each of the dataset are used (26 for P300 data and 40 for spatial data). When using the common model, the subject set is divided into two groups. For the P300 dataset, this gives 13 users and 13 intruders. For the spatial, there is 20 subjects in each group.

For the subject-specific model, all subjects are used for evaluation, which gives 250 and 390 intruder attempts for the P300 and spatial dataset, respectively. The user itself attempts to access the system 10 times, which gives 10 user tests in each experiment. The experiment is then repeated for every subject. The results for the two model layouts can be viewed in table 5.11 and table 5.12.

Classifier	**P300, 26 subjects**			**Spatial, 40 subjects**		
	chans	**TAR**	**TRR**	**chans**	**TAR**	**TRR**
OCSVM dwt	3	0.62	0.55	4	0.18	1.0
OCSVM pca	10	0.13	1.0	2	0.23	0.99
OCSVM emd	14	0.22	0.96	14	0.62	0.43
autoencoder	64	0.77	0.42	64	0.59	0.32
	3	0.82	0.14	4	0.89	0.10
multi-channel	64	0.69	0.49	64	0.72	0.61
	3	0.78	0.23	4	0.88	0.24

Table 5.11: User capacity test for the common model.

Classifier	**P300, 26 subjects**			**Spatail, 40 subjects**		
	chans	**TAR**	**TRR**	**chans**	**TAR**	**TRR**
OCSVM dwt	2	0.27	0.97	2	0.63	0.98
OCSVM pca	6	0.20	1.0	14	0.65	0.94
OCSVM emd	5	0.25	0.99	2	0.71	0.88
autoencoder	64	0.73	0.58	64	0.95	0.92
	2	0.83	0.60	4	0.90	0.97
multi-channel	64	0.92	0.99	64	0.96	0.94
	2	0.94	0.96	2	0.97	0.95

Table 5.12: User capacity test for the subject-specific model.

For the common model, the best result in the P300 dataset is obtained with all channels and the CNN autoencoder. For the spatial dataset, all channels and the multi-channel autoencoder yields the best result. For the subject-specific model,

using 2 channels and the multi-channel autoencoder gives the best result in the spatial dataset with 0.97 TAR and 0.95 TRR. Using all 64 channels yields best result for the P300 data.

As expected, the results decrease when using a larger subset of subjects. However, the decrease in performance when the subject set is twice the size is relatively small. When using only 2 channels, a TAR and TRR of 0.97 and 0.95 for 40 subjects is achieved with the subject-specific model.

5.5.2 Discussion - complete system

Two different methodologies have been proposed for authentication. The results from all experiments show that the subject-specific model is more accurate than the common model.

An advantage of the common model is that this method is not dependent on a correct prediction in the identification layer. As mentioned, this is taken into account by feeding the correct prediction to the subject-specific model. Even though this doesn't affect the results in this work, this is a drawback for the subject-specific model that should be weighted when designing a final system.

When using the common model, the entire system will have to retrain whenever a new user is added or removed, which is time-consuming. The subject-specific model will have to train when a new user is added as well. However this is just the classifier for that specific subject, which takes less time. Besides, the training time for the subject-specific model is shorter, meaning that the process for adding a new subject is faster. When a user is removed, the process is more straightforward for the subject-specific model as that specific classifier can just be deleted from the system and no training is needed.

The different feature selection methods perform relatively even in the experiments. For the common model using the P300 dataset, OC SVM using DWT-based features obtained the best result of all the methods, with TAR and TRR of 0.65 and 0.42. When reducing the training instances to 6, EMD-based features performs best. For the spatial data, PCA-based features are the best methods among the feature extraction methods. The best results obtained with feature extraction is when using EMD-based features and the subject-specific model for spatial data with 0.74 and 0.92 for the TAR and TRR, respectively.

The experiments have shown that it is difficult to find suitable parameters for the OC SVM. Also, as discussed in section 5.3, the values seem to be dependent on the dataset and is difficult to reuse in another part of the data. This is a drawback of using the OC SVM. When using the autoencoders, fewer pre-studies is necessary to

design a good classifier. However, the autoencoders are complex and demand much memory and high computation power. As the results in section 5.5 shows, the time for training the autoencoders are long compared to the OC SVM, where training takes less than a second.

In an effort to reduce training time, the number of instances used for training was tried reduced. Experiments conducted with the subject-specific model and spatial data gives both TAR and TRR of 1.0, using only 18 training instances. Even though the tendency is that the performance is decreasing when reducing the training instances, the performance has been preserved to some extent. It is important to notice that neither the hyper-parameters nor the coefficients for the threshold values are chosen for fewer instances. Hence, better performance may be possible if the values are chosen based on experiments with fewer instances in the prestudies.

Another move made to shorten the training time in the autoencoders, was to use the channels selected by the GA. Even though these channels are not selected explicitly for this model, the performance is very good. For the spatial data using only 2 channels yields TAR and TRR on 0.97 and 0.95, respectively, for the multi-channel autoencoder on 40 subjects. Also in the P300 data, good results are obtained with 0.92 (TAR) and 0.99 (TRR) for 26 subjects, using the multi-channel autoencoder. For both of the autoencoders, the training time has decreased significantly from several minutes to just some seconds. The results indicates that reducing the number of channels is much more effective than reducing the number of training instances both regarding performance and training time.

User-friendliness and real-time aspects are important for a biometric system. Using 24 instances to add a user means that 48 seconds of EEG signals must be recorded. For accessing the system, only one instance is used, i.e., 2 seconds. When looking at similar work presented in table 3.1, 48 seconds is not that much. Other systems uses 3 sessions for training and 1 minute for evaluation [32], 1 minute for training [33], 15 minutes [35] and 39 seconds [37]. In this work, 0.90 for both TAR and TRR is obtained using only 12 instances, which equals to just 24 seconds of EEG data.

The system is implemented asynchronous, meaning that the system is not locked when training the classifier. This means that other users can still access the system while the classifier is training. Thus, the long training time is not critical. This applies both for the common model and the subject-specific model. When using the common model approach, the old classifier is stored so that other subjects can get access when the new model is in training.

Previous studies state that convolution within one channel is more informative than treating the channel as a whole. Comparing the performance of the CNN and the multi-channel autoencoder, one can support this claim as the multi-channel

autoencoder performs better in almost every experiment. However, this autoencoder is a lot more complex, and the number of neurons in each layer is much higher. This results in high computation-time and the training time is almost twice as long in most experiments.

The system is fairly simple, and more layers can be included. As mentioned in the introduction, brain signals are dependent on the mood and stress of the subject, which can lead to challenges for authentication purposes. However, many studies show that such conditions can be detected in the EEG [63, 64, 65]. By adding another layer to detect stress, one can eliminate these factors.

Chapter 6
Conclusion

6.1 Conclusion

The main objective underlying this work was to investigate if EEG-signals can be used for creating a biometric authentication system. Even though other studies have addressed this topic, several aspects have to be improved before commercialisation of such a system. In particular, real-time aspects are not heavily emphasised in previous work.

In this work, a EEG-based biometric system which operated in real-time was realised. The system was implemented as a simple Python server with a MySQL database. Because of unseen events, data acquisition in real-time could not be conducted, and data from two different publicly available databases were used instead. In both datasets, instances were extracted from a resting-state period of the protocols.

Two model layouts were implemented, namely the common model and the subject-specific model. In all the experiments, the subject-specific model provides better performance. As discussed in the previous chapter, a drawback with the subject-specific model is that it relies on correct labelling in the identification layer. On the other hand, this layout is more convenient when it comes to adding and removing users, since the system creates one authentication model for each user.

In addition to testing different authentication methodologies, different methods for feature extraction and classification were tested. For feature extraction DWT, PCA and EMD were investigated. For the classification model OC SVM and two autoencoders built of CNN was implemented. Finding suitable hyper-parameters for the OC SVM turned out to be a challenge. Even though good results (0.97 and 0.98 for TAR and TRR using DWT on spatial data) were obtained when using the GA to find nu and gamma values, the results did not translate well to other sessions of the dataset. A solution suggested in the previous chapter was that the MOOP could also be run in operating time. However, this is a demanding computation process. An

advantage of using DL and autoencoders are that fewer prestudies have to be done. However, the approach proposed in this work still needs some human contribution as a threshold must be chosen for each method.

The GA was also used to select a smaller subset of channels. By reducing the number of channels, one can reduce the training time, thus improve real-time performance. The most frequently selected channels were in the frontal, parietal and central lobes for both of the datasets. Experiments presented in section 5.2 points in the direction that using selected channels can improve, or at least maintain the performance. In all experiments, the results were higher when using a selected subset than when using all channels. Also, as the experiments in section 5.5 shows, using the selected channels gives good performance for the autoencoders as well, even though the channels are not explicitly chosen for those models. TAR and TRR of 0.97 and 0.95 using only 2 channels for 40 subjects are obtained in the spatial dataset with the multi-channel autoencoder.

Another approach to improve the real-time behaviour was to reduce the number of training instances. The results indicates that the performance can be preserved to some extent. The best result was obtained using only 18 instances with both TAR and TRR of 1.0 for 20 subjects. However, the experiments showed that reducing the number of channels is much more effective both regarding performance and training time.

The results in this work give reason to believe that a biometric authentication system based on EEG-signal is indeed possible. The experiments have shown that the performance can be maintained, and in some cases improved, by reducing the number of channels. The system uses between 12 and 48 seconds of EEG data for enrolling and only 2 seconds for accessing the system, which is promising regarding real-time aspects. Concerning other related research, this work has the advantage of using shorter time for recording the EEG data required for enrolment and login. A drawback with the study is that the data is not acquired in real-time. Even though different sessions are used in the pre-studies and the experiments, one can not assert that the system will work for a new set of subjects with data recorded from a different protocol.

6.2 Suggestions for future work

For a final implementation of a real-life system, the data should be collected in real-time. Thus, an experimental protocol must be defined. Based on the experiment in this work, resting-state is a good candidate for the paradigm as good results are achieved. Furthermore, both implementation and setup for the resting-state are simple.

A method for finding the best channels for the autoencoders should be implemented, as this can improve the results even more when using few channels. As discussed in the previous chapter, implementation of the MOOP to find nu and gamma values in operating time may be integrated into the system to improve the performance of the OC SVM.

Other approaches to reduce the time for enrollment can be explored. In this work, instances of 2 seconds are used for both datasets. Instead, one can experiment with smaller instances (for example 1 sec). In this way the system can obtain more instances using less time.

Besides, experiments on other paradigms and datasets should be conducted to ensure generalisation of the methods.

As previously stated, the system is fairly simple and more layers should be included in a final system. A layer to detect whether the subject is stressed or not can help prevent misclassifications. Also, in this work, little focus has been put on data preprocessing and noise reduction in the signals. This topic should be investigated further.

Chapter A
API Documentation

Path	Method	Data parameters	Description

eeg/subject/add	POST	{
		"application_id": 1,
		"name":"Julie"
		}
			Add an user to the system

eeg/subject/remove	POST	{
		"application_id": 1,
		"subject_id":"2"
		}
			Remove an user from the system

eeg/data/save	POST	{
		"application_id": 1,
		"feature_type": "dwt",
		"data": {
		"data": [[[-892.975,-869.8499, ... -888.98]]],
		"target": ["12", ... "12"]
		}
		}
			Save EEG data in the database.

eeg/authentication/training	POST	{
		"application_id": 1,
		"subject_id": 12,
		"model_type": single_model,
		“classifier”: “ocsvm”,
		"feature_type": "dwt"
		}
			Train the authentication model

eeg/authentication/authorize	POST	{
		"predicted_id": 12,
		"model_type": single_model,
		"feature_type": "dwt",
		"data": [[[-892.5, -869.849,..... ,-888.960128]]
		}
			Check if subject is enrolled in the system

eeg/identification/training	POST	{
		"application_id": 1,
		"feature_type": "dwt",
		"classifier": "svm”
		}
			Create ML model for identification layer

eeg/identification/identify	POST	{
		"feature_type": "dwt",
		"data": [[[-892.905,-869.89, ... ,-888.960]]]
		}
			Predict an identify for the subject

Table A.1: Endpoints in server.

References

[1] A. Jain, R. Bolle, and S. Pankanti, *Introduction to Biometrics*, pp. 1–41. Boston, MA: Springer US, 1996.

[2] A. K. Jain, P. Flynn, and A. A. Ross, *Handbook of biometrics*, p. 15. Springer Science & Business Media, 2007.

[3] S. Sanei and J. A. Chambers, *EEG signal processing*, p. 173. John Wiley & Sons, 2013.

[4] J. Haga, "Eeg-based biometric system using discrete wavelet transform and principal component analysis." 5th year specialization project at NTNU, Dec. 2019.

[5] D. T. Stuss and R. T. Knight, *Principles of frontal lobe function*, pp. 11–22. Oxford University Press, 2013.

[6] R. Cooper, J. W. Osselton, and J. C. Shaw, *EEG technology*, pp. 5–10. Butterworth-Heinemann, 2014.

[7] D. Purves, R. Cabeza, S. A. Huettel, K. S. LaBar, M. L. Platt, M. G. Woldorff, and E. M. Brannon, *Cognitive neuroscience*, pp. 1–16. Sunderland: Sinauer Associates, Inc, 2008.

[8] H. H. Jasper, "The ten-twenty electrode system of the international federation," *Electroencephalogr. Clin. Neurophysiol.*, vol. 10, pp. 370–375, 1958.

[9] T. C. Technologies, "10/20 system positioning manual." http://chgd.umich.edu/wp-content/uploads/2014/06/10-20_system_positioning.pdf, 2012.

[10] Q. Gui, M. Blondet, S. Laszlo, and Z. Jin, "A survey on brain biometrics," *ACM Computing Surveys*, vol. 51, pp. 1–38, 02 2019.

[11] M. G. Coles and M. D. Rugg, *Event-related brain potentials: An introduction.*, pp. 173–179. Oxford University Press, 1995.

[12] D. J. Creel, "Visually evoked potentials by donnell j. creel," *Webvision: The Organization of the Retina and Visual System [Internet]*, pp. 1–21, 2016.

[13] M. G. Tramontana and S. R. Hooper, *Assessment issues in child neuropsychology*, p. 347. Springer Science & Business Media, 2013.

[14] I. Daubechies, *Ten lectures on wavelets*, vol. 61, pp. 53–73. Siam, 1992.

[15] C. H. Kim and R. Aggarwal, "Wavelet transforms in power systems. part 1: General introduction to the wavelet transforms," *Power Engineering Journal*, vol. 14, no. 2, pp. 81–87, 2000.

[16] J. C. Lindon, G. E. Tranter, and D. Koppenaal, *Encyclopedia of spectroscopy and spectrometry*. Academic Press, 2016.

[17] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[18] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu, "The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[19] D. Qin, *Power Transmissions: Proceedings of the International Conference on Power Transmissions 2016 (ICPT 2016), Chongqing, PR China, 27-30 October 2016*, pp. 111–112. CRC Press, 2016.

[20] M. Bahoura and J. Rouat, "Wavelet speech enhancement based on the teager energy operator," *IEEE Signal Processing Letters*, vol. 8, pp. 10–12, Jan 2001.

[21] M. J. Katz, "Fractals and the analysis of waveforms," *Computers in biology and medicine*, vol. 18, no. 3, pp. 145–156, 1988.

[22] T. Higuchi, "Approach to an irregular time series on the basis of the fractal theory," *Physica D: Nonlinear Phenomena*, vol. 31, no. 2, pp. 277–283, 1988.

[23] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[24] A. R. Webb, *Statistical pattern recognition*. John Wiley & Sons, 2003.

[25] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*, vol. 112. Springer, 2013.

[26] B. Schölkopf, A. J. Smola, F. Bach, *et al.*, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[27] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[28] M. H. Hassoun *et al.*, *Fundamentals of artificial neural networks*, pp. 58–59. MIT press, 1995.

[29] K. Deb, *Multi-objective optimization using evolutionary algorithms*, vol. 16. John Wiley & Sons, 2001.

[30] M. Mitchell, *An introduction to genetic algorithms*. MIT press, 1998.

[31] N. Sviderskaya and T. Korol'kova, "Genetic features of the spatial organization of the human cerebral cortex," *Neuroscience and Behavioral Physiology*, vol. 25, no. 5, pp. 370–377, 1995.

[32] A. Riera, A. Soria-Frisch, M. Caparrini, C. Grau, and G. Ruffini, "Unobtrusive biometric system based on electroencephalogram analysis," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, no. 1, p. 143728, 2007.

[33] B. Hu, Q. Liu, Q. Zhao, Y. Qi, and H. Peng, "A real-time electroencephalogram (eeg) based individual identification interface for mobile security in ubiquitous environment," in *2011 IEEE Asia-Pacific Services Computing Conference*, pp. 436–441, 2011.

[34] G. Safont, A. Salazar, A. Soriano, and L. Vergara, "Combination of multiple detectors for eeg based biometric identification/authentication," in *2012 IEEE International Carnahan Conference on Security Technology (ICCST)*, pp. 230–236, 2012.

[35] Q. Wu, Y. Zeng, C. Zhang, L. Tong, and B. Yan, "An eeg-based person authentication system with open-set capability combining eye blinking signals," *Sensors*, vol. 18, no. 2, p. 335, 2018.

[36] C. R. Hema, M. P. Paulraj, and H. Kaur, "Brain signatures: A modality for biometric authentication," in *2008 International Conference on Electronic Design*, pp. 1–4, 2008.

[37] L. A. Moctezuma and M. Molinas, "Multi-objective optimization for eeg channel selection and accurate intruder detection in an eeg-based subject identification system," *Scientific Reports*, vol. 10, no. 1, pp. 1–12, 2020.

[38] S. Marcel and J. d. R. Millán, "Person authentication using brainwaves (eeg) and maximum a posteriori model adaptation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 4, pp. 743–752, 2007.

[39] D. La Rocca, P. Campisi, and G. Scarano, "Eeg biometrics for individual recognition in resting state with closed eyes," in *2012 BIOSIG - Proceedings of the International Conference of Biometrics Special Interest Group (BIOSIG)*, pp. 1–12, Sep. 2012.

[40] K. Brigham and B. V. K. V. Kumar, "Subject identification from electroencephalogram (eeg) signals during imagined speech," in *2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pp. 1–8, Sep. 2010.

[41] L. A. Moctezuma and M. Molinas, "Subject identification from low-density eeg-recordings of resting-states: A study of feature extraction and classification," in *Future of Information and Communication Conference*, pp. 830–846, Springer, 2019.

[42] L. Moctezuma, A. Torres-García, L. Villaseñor-Pineda, and M. Carrillo, "Subjects identification using eeg-recorded imagined speech," *Expert Systems with Applications*, vol. 118, pp. 201–208, 03 2019.

[43] L. A. Moctezuma, A. A. Torres-García, L. Villaseñor-Pineda, and M. Carrillo, "Subjects identification using eeg-recorded imagined speech," *Expert Systems with Applications*, vol. 118, pp. 201–208, 2019.

[44] Y. Bai, Z. Zhang, and D. Ming, "Feature selection and channel optimization for biometric identification based on visual evoked potentials," in *2014 19th International Conference on Digital Signal Processing*, pp. 772–776, IEEE, 2014.

[45] D. Rodrigues, G. F. Silva, J. P. Papa, A. N. Marana, and X.-S. Yang, "Eeg-based person identification through binary flower pollination algorithm," *Expert Systems with Applications*, vol. 62, pp. 81–90, 2016.

[46] Q. Gui, Z. Jin, and W. Xu, "Exploring eeg-based biometrics for user identification and authentication," in *2014 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*, pp. 1–6, Dec 2014.

[47] F. Lotte, L. Bougrain, A. Cichocki, M. Clerc, M. Congedo, A. Rakotomamonjy, and F. Yger, "A review of classification algorithms for eeg-based brain–computer interfaces: a 10 year update," *Journal of neural engineering*, vol. 15, no. 3, p. 031005, 2018.

[48] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball, "Deep learning with convolutional neural networks for eeg decoding and visualization," *Human brain mapping*, vol. 38, no. 11, pp. 5391–5420, 2017.

[49] Z. Tang, C. Li, and S. Sun, "Single-trial eeg classification of motor imagery using deep convolutional neural networks," *Optik*, vol. 130, pp. 11–18, 2017.

[50] A. Antoniades, L. Spyrou, C. C. Took, and S. Sanei, "Deep learning for epileptic intracranial eeg data," in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, IEEE, 2016.

[51] J. Liang, R. Lu, C. Zhang, and F. Wang, "Predicting seizures from electroencephalography recordings: a knowledge transfer strategy," in *2016 IEEE International Conference on Healthcare Informatics (ICHI)*, pp. 184–191, IEEE, 2016.

[52] P. Bashivan, I. Rish, M. Yeasin, and N. Codella, "Learning representations from eeg with deep recurrent-convolutional neural networks," *arXiv preprint arXiv:1511.06448*, 2015.

[53] M. Hajinoroozi, Z. Mao, T.-P. Jung, C.-T. Lin, and Y. Huang, "Eeg-based prediction of driver's cognitive performance by deep convolutional neural network," *Signal Processing: Image Communication*, vol. 47, pp. 549–555, 2016.

[54] S. Opałka, B. Stasiak, D. Szajerman, and A. Wojciechowski, "Multi-channel convolutional neural networks architecture feeding for effective eeg mental tasks classification," *Sensors*, vol. 18, no. 10, p. 3451, 2018.

[55] S. Chakraborty, S. Aich, M.-i. Joo, M. Sain, and H.-C. Kim, "A multichannel convolutional neural network architecture for the detection of the state of mind using physiological signals from wearable devices," *Journal of healthcare engineering*, vol. 2019, 2019.

[56] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces," *Journal of neural engineering*, vol. 15, no. 5, p. 056013, 2018.

[57] H. Morioka, A. Kanemura, S. Morimoto, T. Yoshioka, S. Oba, M. Kawanabe, and S. Ishii, "Decoding spatial attention by using cortical currents estimated from electroencephalography with near-infrared spectroscopy prior information," *Neuroimage*, vol. 90, pp. 128–139, 2014.

[58] yieldwerx, "Part average testing (pat) tutorial," 2019. http://yieldwerx.com/part-average-testing-pat-tutorial/.

[59] G. Upton and I. Cook, *Understanding statistics*. Oxford University Press, 1996.

[60] S. Opałka, B. Stasiak, D. Szajerman, and A. Wojciechowski, "Multi-channel convolutional neural networks architecture feeding for effective eeg mental tasks classification," *Sensors*, vol. 18, no. 10, p. 3451, 2018.

[61] S. Premkumar, "Biometric identification system using eeg signals," Master's thesis, NTNU, 2020.

[62] M. Själander, M. Jahre, G. Tufte, and N. Reissmann, "EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure," 2019.

[63] J. Knaus, R. Wiese, and U. Janßen, "The processing of word stress: Eeg studies on task-related components," in *Proceedings of the 16th international conference of phonetic sciences*, pp. 709–712, Saarbrücken Germany, 2007.

[64] N. Sulaiman, M. N. Taib, S. Lias, Z. H. Murat, S. A. M. Aris, and N. H. A. Hamid, "Eeg-based stress features using spectral centroids technique and k-nearest neighbor classifier," in *2011 UkSim 13th International Conference on Computer Modelling and Simulation*, pp. 69–74, IEEE, 2011.

[65] F. Al-Shargie, T. B. Tang, N. Badruddin, and M. Kiguchi, "Mental stress quantification using eeg signals," in *International Conference for Innovation in Biomedical Engineering and Life Sciences*, pp. 15–19, Springer, 2015.

NTNU

Norwegian University of
Science and Technology