Technical University of Denmark

02466: Course Project - Bachelor in Artificial Intelligence and Data

Ensuring data quality of Electroencephalography (EEG) recordings through anomaly detection using transfer learning

Authors:
Johannes Boe Reiche s175549
Mads Christian Berggrein Andersen s173934
Villads Tristan Stokbro s175548
Andreas Råskov Madsen 183901

Academic advisor:
Lars Kai Hansen, Professor at DTU Compute.

March 22, 2025

Abstract

Machine learning models have accomplished impressive results in a wide range of application areas, and this project will investigate how machine learning techniques can be utilized to classify the clinical usability of electroencephalography (EEG) signals. By creating a latent representation of EEG-spectrograms with the use of a convolutional neural network and comparing the quality of the new representation based on classification performance, we wish to unravel the potential of direct transfer learning of the well known CNN "VGG16" first presented in [9]. Both EEG-spectrograms and the latent representation are trained on a range of different machine learning models, and evaluated against a baseline model, which guesses on the largest class ("Usable for clincal purposes"). Furthermore this project investigates the potential of these representation using visualizations techniques such as t-SNE and PCA to explain what grouping the classifiers may identify. In addition saliency maps investigates the evidence that a VGG16 finds when making predictions on the clinical usability of a EEG-spectrogram, to further explain its feature representation. Lastly the generated latent representations is explored further with unsupervised clustering. With a varying degree of certainty this project found that some stereotype structural patterns can be identified by a Gaussian Mixture Model applied on feature vectors from a finetuned VGG16. We found that machine learning models including Support Vector Machine and Random Forest on both representations were able to beat a baseline model. Caution of these results have to be taken due to group structures in the data. Furthermore did the evidence presented in the project suggest that the latent representation of EEG-signals did not pose as a better data representation for classification purposes in this project. However the quality of the labels used for classification can be discussed. Some meaningful features seemed to be emphasized by a tuned VGG16 based on the saliency maps produced.

Contents

1 Introduction

Epilepsy is one of the most frequent neurological disorders worldwide and the cumulative risk of developing epilepsy by the age of 80 ranges from 1.4-3.3 % [8]. In technologically advanced countries epilepsy can be diagnosed with the use of both expensive medical equipment and well-educated staff and the right anti-epileptic medicine can be prescribed. However it is estimated that around 80% of the people with the disorder are from low-middle income countries [24]. Additionally, almost three quarters of the affected people are not diagnosed and treated in developing countries according to the World Health Organization (WHO)[24], and in certain countries up to 90%. WHO estimates that 70 % of the people who gets a diagnosis can be treated with medicine that is economically feasible even in developing countries. To solve this problem WHO has started a program with the aim of reducing the treatment gap.

In a research project by Williams *et al.* [17] it was concluded that a cheap EEG sensor-cap developed by The Technical University of Denmark can be used to diagnose epilepsy by transferring EEG data to a cloud based system, and distributing it to experts for classification. Brain-Capture is a startup company that seeks to bring this from an academic research project to a product that can be rolled out to a big part of the world's population in developing countries. In order to collect EEG's that are suited for clinical purposes, onsite quality inspection of the signal is needed. Since the target of this solution is rural areas without any former access to EEG-equipment, the medical staff will not be properly educated to handle EEG signals. To assist the medical staff in detecting errors in the recordings, machine learning solutions are needed.

1.1 State of the art research

There exists a wide range of proved and promising approaches to identifying abnormalities in medical data and in this section several articles will be reviewed in order to introduce where some of the ideas used in this project stem from, and what can be achieved with these approaches. The complexity of the approaches varies and includes simple similarity techniques in which abnormalities are identified based on a simple threshold and more complex methods including Temporal Convolutional Neural Networks (TCNN) combined with Gaussian Mixture Models (GMM). The first mentioned approach comes from a research article by Chen et al [18] which is a rather simple case study that investigates the detection accuracy of abnormal EEG-observations based on different similarity measures. They extracted the power spectrum and quantified the similarity by measuring the Hellinger Distance (HD), Bhattacharyya Distance (BD) amongst others. From the optimal threshold for decision making they accomplished an accuracy of 96.67 % on anomaly EEG detection on the Bern-Barcelona EEG data set that consisted of normal and undefined abnormal recordings. Another approach proposed by Frølich *et al.* revolves around a multinomial regression model to automatically detect artifact types in EEG signals [7]. The research team extracted 65 features from the EEG signals characterized as either spatial, spectral or temporal and then used forward selection to identify which features that were the most informative. They found that a subset of 14 features was optimal and achieved a balanced accuracy of more than 80% when classifying the six artefacts types present in the dataset. This accuracy was obtained on both of their datasets tested within subjects.

However the approach was less accurate on subjects between studies.

A general framework for detecting collective anomalies (a sequence of data points, which together are abnormal) in time series with the help of TCNN's and a Gaussian Mixture Model (GMM) was proposed by Liu *et al.* [15]. In the study a TCNN was utilized to extract salient features, that could be used in the subsequent analysis of the signals and classified using Bayesian inference of a GMM. The framework was tested under two rather different set of circumstances, one in which EEG data was investigated, and another with anomalous electrical current (Arc Fault mode vs normal behaviour). The model achieved a F1 score of 99.54% when classifying periods with epileptic seizures from regular periods of brain activity (i.e resting state with or without closed eyes). When classifying default operation of an air conditioner vs Arc Fault operation the framework achieved an F1 score of 99.66%. It is unknown how well the methods generalized to other data sets that it had not been trained on.

Independent Component Analysis (ICA) has also shown to be effective in extracting features from the signal, specifically the independent sources both neural and non-neural that contributes to the signal. A variety of ICA-classifiers have been developed in order to detect unwanted sources in the signal. Among those the "ICLabel" created by Luca Pion-Tonachini, Ken Kreutz-Delgado and Scott Makeig in 2019, which comes as an extension to the MATLAB toolbox "EEGLAB" . The classifier is trained using an ANN on 200.000 labeled IC's from a crowd sourced database, and tested on 130 IC's from different datasets labeled by two experts. Pion-Tonachini2019ICLabel:Website For binary classification of the classes "Brain" and "Other, the classifier receives a balanced accuracy of 0.85, and for five classes with "Line/Channel-noise", "Muscle/Heart", and "Eye" added an accuracy of 0.65. The "ICLabel" performs similarly but slightly better than the other classifiers based on accuracy, but is 13 times faster than the second best with a processing time of 0.1 seconds.

A pilot study using a deep Convolutional Neural Network (CNN) for abnormality detection on spectrograms from the BC data set, which is introduced in section 2.3, proved to be able to detect added white noise with 0.978% accuracy thus proving the potential of the method that was used in the report [23]. A project made on University of Surrey have already proven that a CNN can be used to detect Interictal epileptic discharges in EEG data with 87.51% accuracy from a dataset labeled by human experts [20].

Transfer learning using pretrained weights of deep CNN's trained on large databases of images like Imagenet, has been utilised in computer aided vision (CAD) tasks in which labeled data is sparse. The method builds on the hierarchical feature learning of a CNN, utilising general feature maps relevant for transformation invariant object detection and segmentation. A study investigating the use of transfer learning in thoraco-abdominal lymph node (LN) detection and interstitial lung disease (ILD), showed that for multi-class categorization of a smaller data set, 1000 labeled samples in the case of ILD-classification, transfer learning showed the most promising results with an accuracy of 54 % on 6 classes. However for LN classification, which is more suitable for data augmentation and therefore more data is available, the advantage of using transfer learning is less significant. [12]

1.2 Purpose of the project

The overall goal in the long term is to create a framework that can assist both medical staff and personnel in developing countries when EEG-measurements are performed on patients to ensure the quality of the EEG-session with the intend of diagnosing epilepsy. The primary focus of this project is therefore to analyze brain activity measured with EEG-equipment and detect anomalous signals. Here an anomalous EEG-signal is defined as a deviation from EEG-signals usable for clinical trials. Another important aspect of the project is to figure out how the models used to detect anomalies in the EEG signals can be interpreted in a meaningful way, such that they provide specific guidance in a manner such that doctors or other personnel can translate the outputs from the model directly into actions. Put in another way: The model's explainability is important.

1.2.1 Course of action

We will try to classify time-windows of EEG-recordings from a data-set provided by Brain-Capture consisting of 126 EEG-recordings, based on the label 'is-usable' stating whether the recordings are fit for clinical purposes or not. The labels are not annotated on the specific time-windows of the recording but on the full recordings. We will test two data representations; spectrograms of two-minute windows of EEG-recordings and a feature representation made by a pretrained Convolutional Neural Network VGG16 of the same spectrograms. These two representations will first be visualised through dimensional reduction methods, and then tested on a range of machine learning models. Their test accuracy's will be compared in order to evaluate the potential of each representation. Saliency maps for a retrained VGG16 will be visualized together with raw EEG signals as a qualitative method to check whether transfer learning can be utilized on the BC data set to learn certain characteristics of EEG signals on limited data. At last we will try to cluster the EEG windows that the retrained VGG16 correctly classifies as being non-usable, in order to find any common anomalies.

1.2.2 Research questions

This project will try to answer the following research questions:

1. Is it possible to make a classifier that outperforms a baseline[1], to classify unwanted disturbances in fragments of the signal, based on the 'is-usable' label of the BrainCapture EGG-recordings?
2. Can we gain a data representation superior to spectrograms of EEG-signals through transfer learning?
3. Is it possible to gain an understanding of how a CNN percieves an EEG-spectrogram and which features are extracted through saliency mapping?
4. Will it be possible to identify characteristics of the correctly predicted 'not_usable' EEG-windows with unsupervised clustering?

The hypotheses for the research questions are presented in the following:

1) It is possible to make a classifier as described in research question 1, that significantly outperforms a baseline.

2) By propagating the EEG-data through the CNN our hypothesis is that the latent representation successfully captures useful features and discards redundant information. Consequently, we hypothesize that the classification models can achieve a higher accuracy on the latent representation than directly on spectrograms, when predicting the usability of a signal.

3) Through saliency mapping it is possible to explain the evidence a CNN finds for predicting an EEG-spectrogram to be 'not_usable'.

4) Artefacts that cause signals to be anomalous stem from the same distinct sources, such that an unsupervised clustering method is able to create separate clusters for each artefact type.

[1]A baseline, in this context, is a classification model, that predicts every observation to the most frequent class.

2 Data

An introduction to epilepsy and electroencephalography (EEG) will be given to ensure that the reader is familiar with the concepts and terms used later on. Classification of epilepsy in EEG is beyond the scope of this project, however this is at the core of Brain-Capture's mission, and the data used in this project will contain signs of epilepsy. It is therefore included in the following description.

2.1 Epilepsy and Electroencephalography

The term epilepsy accounts for a group of neurological disorders that are all characterized by recurring periods with symptoms varying from uncontrolled shaking movements to loss of consciousness [8]. These symptoms are caused by abnormally excessive or synchronous neural activity in the brain and these periods are also known as epileptic seizures. These seizures are unprovoked and can therefore be challenging to live with. Even though epileptic seizures in general are not fatal, injuries can occur with variation of severity ranging from bruises to death in rare cases.
To diagnose epilepsy in a patient is not straight forward, and a diagnosis is often based on both symptoms and assistance from one or several medical procedures. The medical procedures used when diagnosing includes electroencephalography (EEG), computed tomography scan (CT) and Magnetic resonance imaging (MRI) [26].
Electroencephalography (EEG) is a way to measure the electrical activities in the brain by placing multiple electrodes on the scalp of a patient. This allows for a wide range of scientific research as it is a relatively cheap and non invasive way of measuring brain activity. In medical treatments it is used to diagnose sleep patterns and to diagnose epileptic seizures. The EEG signal have some clear downsides. The signal originates from the action potential groups of neurons firing in the brain, and the electromagnetic potential that is measurable outside the scalp is therefore not strong. As a results EEG data is generally noisy and easily disturbed by surroundings or other bodily functions, such as eye blinks, mussel tensions, heart rhyme. EEG signals are usually based on the brain firing in a rhythm (limited frequencies span) but disorders such as an epileptic seizure will disturb this rhythm. EEG is also individual for each person making it hard to generalize a classifier across subjects. EEG signals also change depending on the age [4], and especially in infants can the EEG signal be a lot more powerful due to the underdeveloped cranium.

In order to get a visual example of epilepsy in an EEG recording with 18 electrodes (channels) is illustrated below in figure 2.1.1:

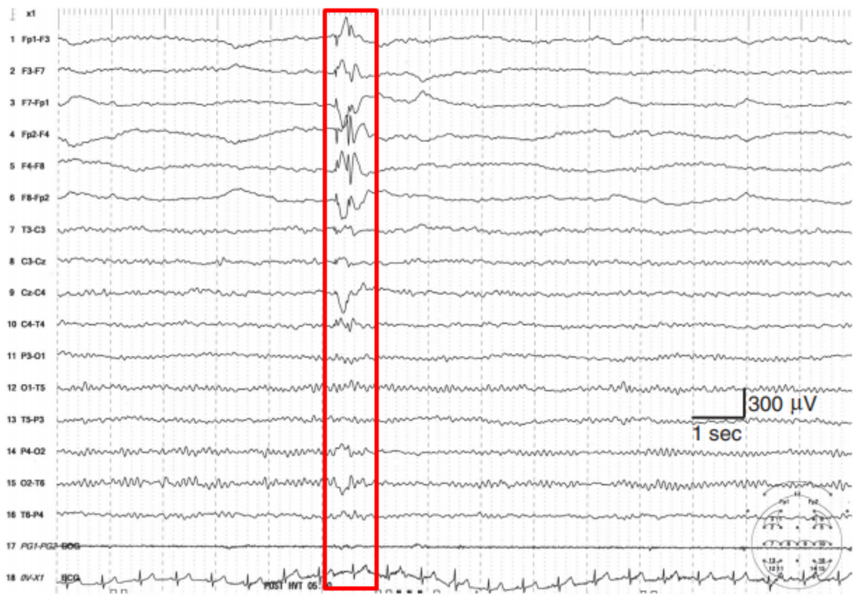


Figure 2.1.1: Visualization of Eleptiform Discarge in frontal areas. Plot taken from [10].

The most common way of setting up electrodes for EEG is known as the 10-20 system with 21 electrode. It is based on placing one electrode on the nasion, one on the inion, then separating the distance between those points in the following intervals 10%, 20%, 20%, 20%, 10% hence the name. See figure 2.1.2 for precise setup, and for the setup used in the. Another system called the 10-10 system has 71 electrodes. The 10-10 system shares the same electrodes as the 10-20 system but have additional electrodes placed for every 10 percent interval.

Due to the low signal to noise ratio, EEG signals are generated by comparing electrodes. However a range of different montage sequences can be used to compare the electrodes. The most common is the bipolar montage comparing each electrode to its neighbor, starting either form the top going down along the 10-20 lines, or stating from the left to right. Other montages includes comparing to the average over all the electrode, or comparing to a single reference electrode.

Traditionally brain rhythms would be represented as different brainwaves, brainwaves are defined as the amplitude of measurement at a given frequency range. There is not an exact academic consensus of the exact bandwidth, according to Saeid Sanei and Jonathon Chambers[4] there are the following brainwaves: δ waves 0.5-4 Hz are primarily found in deep sleep. θ waves 4-7.5 Hz are associated with unconsciousness deep meditation or arousal. α waves 8-13 Hz are associated to relaxation and deep concentrations. β waves 14-26 Hz is associated with high brain activity, both in the state of high focus or in panic state. γ waves are above 30 Hz but are rarely seen in humans, but can sometimes be used to diagnose brain disorders [4].

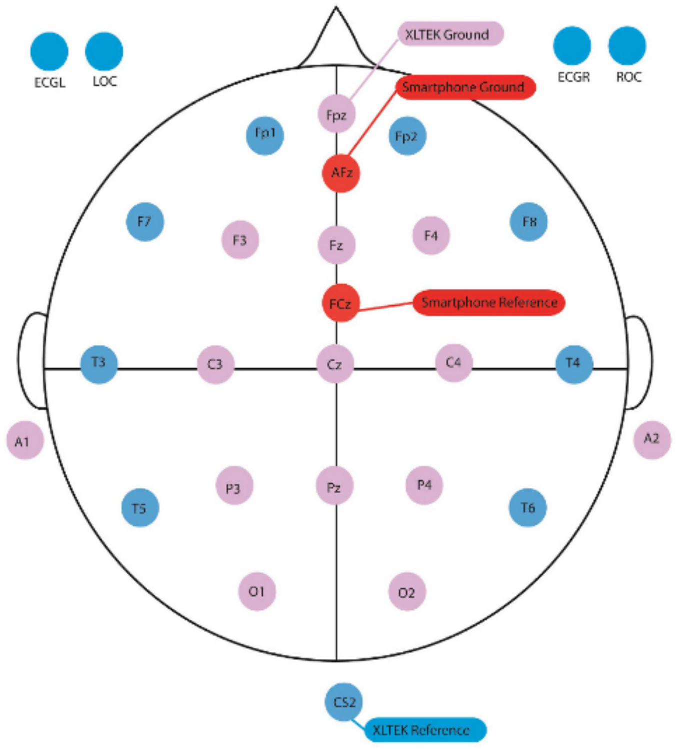


Figure 2.1.2: Picture of EEG channel setup/montage. Blue and pink is part of 10-20 system. Pink is the BC data montage. Red is reference point use for prepossessing (Not part of our data). figure credit [14]

2.2 Visualization of artefacts in EEG signals

As mentioned in the introduction, one of the goals of this report is to be able to explain the underlying cause for why a signal being classified as 'not-usable'. Some of these underlying causes are referred to as EEG artefacts in medical literature. These artefacts have distinct characteristics, which is used by neurologists to differentiate between them, and they could potentially be learned by the machine learning models. These characteristics include frequency range, amplitude and amount of electrodes that are influenced.
This section will provide visualizations of some of the most common artefacts found in EEG-signals. The EEG-artefacts are usually divided into neural and non-neural. Four examples of non-neural artefacts are visualized in figure 2.2.7. The y-axis of the following plots 2.1.1 & 2.2.7 show the amplitudes (measured in μV) of each of the different electrodes/channels that are placed on the scalp of the test subject. The x-axis show the time in seconds. It can be seen from the plots that the artefacts have distinctive patterns and that the signals are highly irregular in the periods in which the artefacts occur.

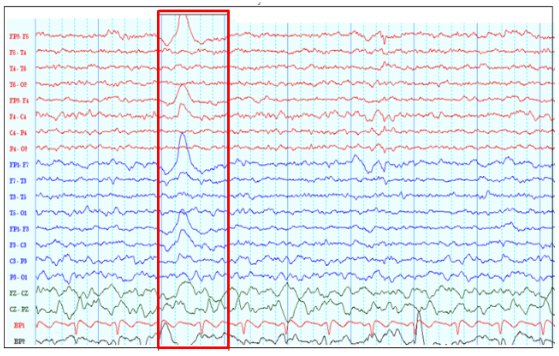


Figure 2.2.3: Eye Blink

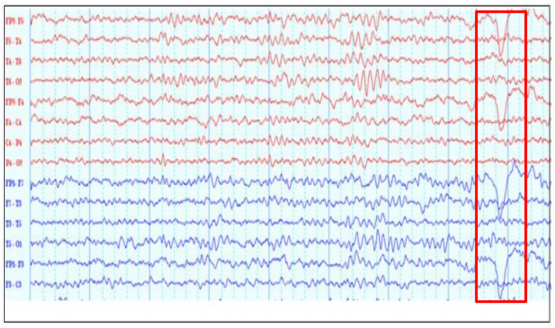


Figure 2.2.4: Eye Movement

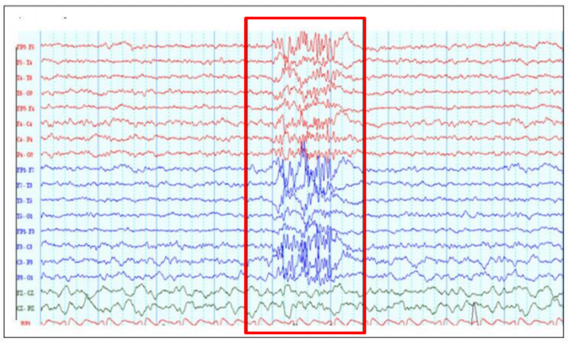


Figure 2.2.5: Muscle movement

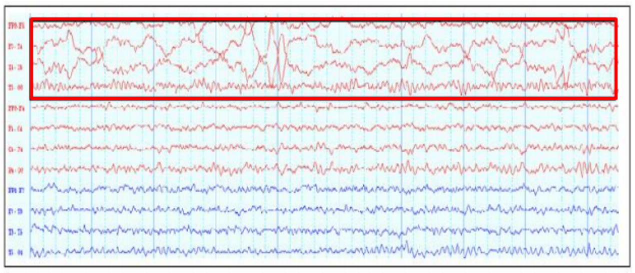


Figure 2.2.6: Electrode pop

Figure 2.2.7: Credits to Avinash Tandle and coworkers for the visualizations above [13].

2.3 The Brain Capture data set

The Brain Capture (BC) data set is provided by the company Brain-Capture and it contains data from a EEG-gathering session in Republic of Guinea (2018-2019). The data set contains a total of 512 EEG recordings measured on individuals. Only 126 of the 512 recordings are sufficiently labelled for our supervised purposes, therefore 398 recordings are disregarded. This brings the amount of EEG-recordings labelled as clinically usable to 100, and non-usable recordings to 26. The 126 files that is used is a subset of the data set used in [19]. Information on the data gathering process has been found in [19] (the references is scientific articles written in cooperation with Brain-Capture).
An important note on this data set is that it has been gathered with a 14-electrode EasyCap that is significantly more affordable (cost: 300 USD) than regular EEG caps. The electrodes were placed according to the international 10-20 system and positioned at F3, C3, P3, O1, F4, C4, P4, O2, Fz, Cz, Pz, Fpz, A1, and A2, where FCz serves as the reference electrode and AFz as the ground electrode see montage in figure 2.1.2. The Easycap is used with android tablet and 'The Smartphone Brain Scanner-2' software, which makes the whole setup a viable option in poor regions of the world. The data was gathered with a sampling rate of 128Hz.

The data was first gathered with the purpose of evaluating the quality of the recordings compared to traditional EEG equipment. The data for each recording is stored in one to three European Data Format files (.edf). The length of the 126 recording used in the project can be seen in table 1.

Length of recordings	min	25 % -quantile	median	75 % -Quantile	max
all	5.05	54.35	59.2	60.38	87.38
Is usable	13.68	54.85	59.35	60.42	87.38
Not usable	5.05	50.86	57.28	60.18	65.5

Table 1: Length of recordings in minutes

Each recording happened while the test subject was lying in a bed face upward, and with instructions of having a minimum of movement with closed eyes for the whole duration of the recording. Some recordings include periods of drowsiness and sleep. Along with each .edf file is an evaluation of the signal. Each evaluation contains meta data and an evaluation with annotations from a neurologists (a few recordings have more than one neurologist assigned). It is important to stress, that these annotations belongs to the whole file. 11 different neurologists have been used to annotate the 126 files used. Each evaluation includes a range of possible annotation (including artefacts like Epileptiform Discharges Focality Frontal etc.), a quality score (1-10) from the neurologist who looked at it, a binary score telling if the data is high enough quality to be used for medical purposes, recommendation on further treatment/measurement. An elaborate list of what is included can be found in appendix [8.2]. For a more detailed explanation of test subjects and the how the data gathering was performed see [19] and [14]. Another important remark is that some individuals are measured twice separated with a variable interval. The meta data is as summarized in table 2 and 3.

Age	min	25 % -quantile	median	75 % -Quantile	max	Under the age of 2
all	0.83	10	18.5	31.5	66	5
"usable"	1	12.5	20	34	65	2
"not_usable"	0.833	4.5	13	21	66	3

Table 2: Age of test subjects in years and number of test subject under the age of 2

Sex	Man	Woman
all	66	60
"usable"	54	46
"not_usable"	12	14

Table 3: Sex of test subjects

3 Theory

3.1 Spectrograms

A spectrogram is a visual representation of a time-series that displays the amplitude of the frequency components of a signal as it varies through time. Spectrograms are usually visualized as heatmaps such that the amplitude of the frequencies can be depicted as the intensity of a color.

In contrast to spectrograms, a normal waveform representation of a signal, such as a regular EEG-montage, only depicts the amplitude and how it varies through time. If such a waveform signal consists of only a single frequency component, one could get the signal frequency by looking at the period of the signal. However complex signals (such as EEG) usually contains several frequency components which look inseparable in a waveform visualization. A spectrogram is generated with a Fast Fourier transformation (FFT), in which a signal is decomposed into its frequency components. This separation can be very beneficial when looking at EEG-signals because, as is mentioned in 2.1, different frequency spectrums stem from different types of brain activity.

In detail the generation of spectrograms happens by dividing the time-domain signal into time-segments (usually with some overlap) and then calculating the magnitude of frequency components in a signal with FFT. Each time-segment of signal is then converted into power spectrum, which is subsequently turned into a vertical bin, which represents the power of different frequencies in a given period of time. All these bins are then put next to each other to generate the spectrogram. In figure 3.1.8 visualization of a raw EEG-window with 4 channels can be seen and its corresponding spectrograms in 3.1.9.

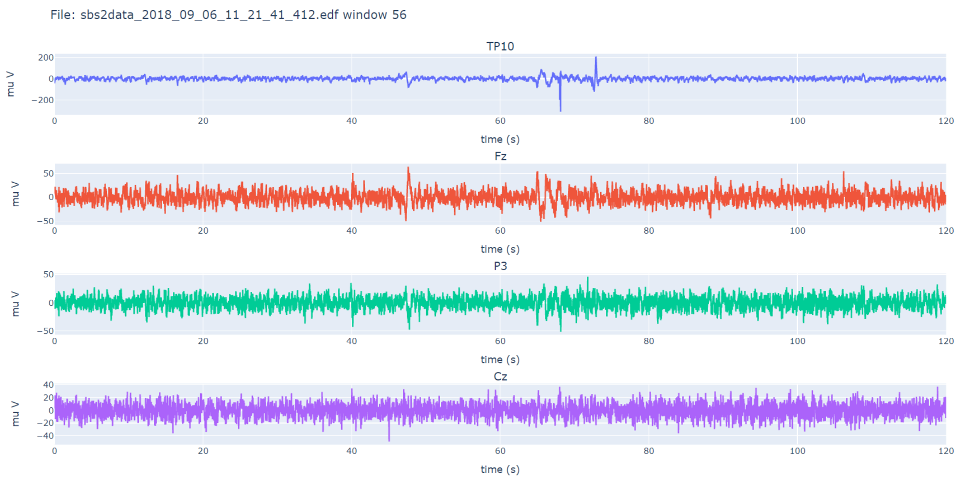


Figure 3.1.8: Preprocessed EEG signal

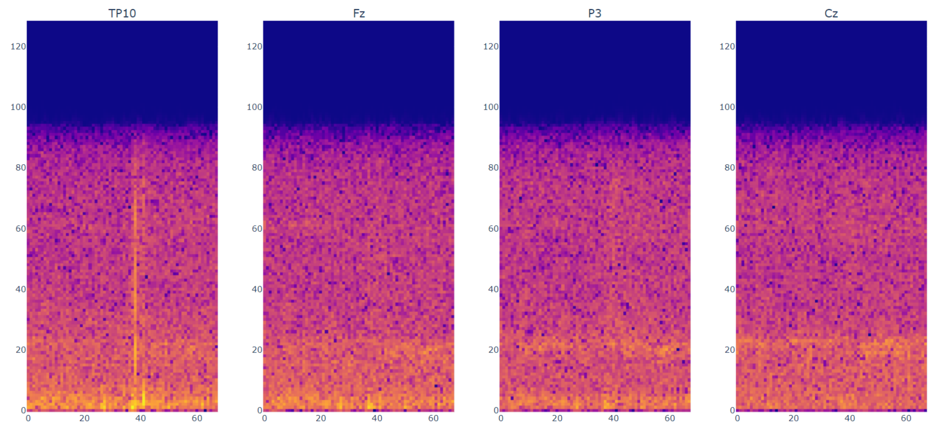


Figure 3.1.9: Example of how EEG data is transformed to spectrograms for 4 channels

3.2 Latent Space and Convolutional Neural Networks

3.2.1 Introduction to latent space

The latent space is a term that refers to a space wherein latent variables lie. Latent variables are hidden variables that cannot directly be observed, but they can be inferred from observable variables. Latent variables are usually designed such that they contain meaningful and sometimes abstract information. A latent representation of a picture may instead of containing raw pixel values contain information about the content of the picture instead.

With a raw pixel image representation it is difficult to train a model to identify objects and generalize well to unseen data points. This is due to the fact that changes that we as humans perceive as minor such as change of position, rotation, change of background etc., may drastically change the pixel values. If one uses a latent representation instead, rotations or similar transformations may only result in lesser changes to the data representation, which may allow machine learning models to generalize better. The generation of latent variable is usually accompanied by a dimensionality reduction.
When a latent representation of a data set has been generated, it is then possible to extrapolate from the latent space, and latent variables that are close to each other have similar features and are therefore usually considered to belong to the same class. However in the process of generating a latent data representation is an information extraction and as a consequence there is a loss of information. The information that persists may or may not be the information that is most valuable, which can potentially make the representation be useless.
The following sections will discuss and argue for why it seems reasonable and possibly beneficial to use a pretrained VGG16 to compress our EEG-signals and use the resulting latent representation of the EEG-signals for subsequent analysis and classification.

3.2.2 Manual vs Automatic feature extraction

A core concept in machine learning is Representation Learning also known as Feature Learning. The term describes a set of techniques that can be used to identify the representation needed to perform feature detection or classification tasks from raw input data. This is quite an important part of machine learning, and the performance of machine learning models is heavily dependent on the data representation [21]. The data representation can either be chosen manually or learned automatically. An example of the manual task of feature engineering is mentioned in the 1.1 section. Here [7] investigated classification of EEG-artefacts by first extraction 65 features either in the spatial, temporal, or spectral domain. This form of manual extraction of features requires a lot of expert knowledge, forethought and effort to perform. Further more this compressed signal representation is not guaranteed to contain the relevant information, and it is possible that valuable information is lost due to human bias or lack of insight. One of the grand benefits of using a CNN is that this feature learning is performed automatically during training of a CNN, and the CNN might learn underlying and more general explanatory factors that might be beyond human perception. It may even be possible to transfer learned features from one task to another. The concept of transfer learning is the main reason for why a pretrained VGG16 is used to generate feature vectors. In the next section it will be explained how the feature extraction works in a CNN and afterwards a description of the VGG16 network.

3.3 Inner workings of a CNN

This section is written with inspiration drawn from Michael Nielsens Book on Neural Networks chapter 6 [27] and Deep Learning Book chapter 6 and 9 by Ian Goodfellow, Yoshua Bengio and Aaron Courville [11].

3.3.1 Forward Propagation

This section will dive into how a CNN works and how data is represented and compressed throughout the layers. A CNN is a class of neural networks that contain at least one convolutional layer. A convolutional layer is defined as a hidden layer that uses an operation called convolution to process information. A convolution is an operation on two functions that generate a third, which express how the shape of the first function is reshaped by the second [22]. When dealing with convolutional networks the first argument to the convolution is the input to the CNN and the second argument the kernel. The result of such a convolution is referred to as a feature map or sometimes activation map. The equation of a discrete convolutions C (in two dimensions) is in [11] defined as

$$C(i,j) = (K \cdot I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n). \tag{3.3.1}$$

Here I refers to the image input, and K the kernel. (i,j) is the position in the center of the region of the image, that is currently being convolved. m and n translates to how many neighbouring pixel (relative to the (i,j) that is used to generate the output. n,m is dictated by the kernel dimensions and if f is the length and width of the kernel, both m, and n can be calculated by

$$m = n = \frac{(f-1)}{2}. \tag{3.3.2}$$

To match the definition of a convolution, the kernel needs to be flipped relative to the input. However the flip is not considered of importance, and many machine learning libraries implement the almost identical function cross-correlation Cr, which is the same equations but without a flipped kernel

$$Cr(i,j) = (K \cdot I)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n). \tag{3.3.3}$$

The convolution can be seen as an element-wise multiplication, where the kernels are put on top on the input. The convolution equation can be extended to 3D, As it is visualized with a yellow box in 3.3.10, the kernel is placed on top of the input, and then all values from the input and kernel in the same position defined as (height,width,channel) are multiplied and summed altogether.

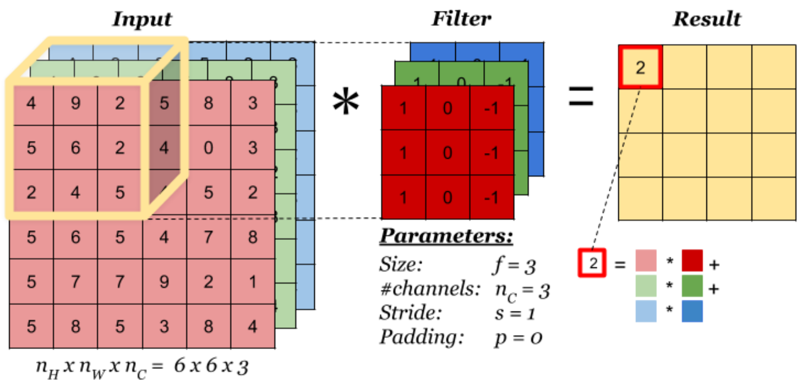


Figure 3.3.10: Convolution operation - Picture taken from [https://www.zybuluo.com/hongchenzimo/note/1086311]

The trainable parameters in a convolutional layer are the kernel weights and then an additional bias value for each kernel. To each convolutional layer multiple kernels usually exists, and intuitively these kernels learn to look for different aspect within a photo. Each kernel slides over an input and generates the feature maps. How many pixels a filter stride between each convolution is a hyperparameter that has to be chosen. A large stride results in a larger dimensionality reduction, which is computationally a benefit, but information is lost. To avoid any dimensionality reduction at all padding is needed, and one popular way of doing so is by adding one or several 0-values to the edges of an input. The feature map dimension can be calculated as (assuming input is square i.e $n_H = n_W = n$)

$$(n, n, n_C) \cdot (f, f, n_C) = (\frac{n-f+2p}{s}, \frac{n-f+2p}{s}, n_k).$$

The first parenthesis from the left contains the input dimension, second the kernel dimension and the third the feature map dimensions. n_k is the number of kernels in convolutional layer indexed by k. The other variables are described in figure 3.3.10, except n_k which is the number of kernels in convolutional layer indexed by k.
There are several benefits of using convolutions. First off, a benefit is that there is sparse interactions. This means that neurons in a feature map are generated from only a limited amount of adjacent neurons taken from the previous layer. The result of sparse interactions is that large inputs can be processed with only a few weights and it requires fewer operations.
Secondly convolutional layers use parameter sharing, which mean that all neurons in a feature map are generated with the same kernel. This means that every weight in the kernel is used multiple times, which is not the case in dense layers, where each weight is only used once to compute an output. Parameter sharing in convolutional layers reduces the trainable parameters drastically compared to dense layers, the results of this is that deeper networks are feasible.
A third advantage that convolutional layers have is called equivariant representations, which means that a change in the input will have the same effect in the output. Say an object in an image is shifted from one side of an image to another, the representation of this object in the output will also be shifted.

The way a convolution layer operate can be divided into three processes. First, the convolutions are performed and produces linear activations, which in the second process is parsed through a non-linear activation function. The non-linear activation function is important as it makes the CNN more expressive (non-linear). There exists several activation functions, and the most widely used is a rectified linear activation (ReLU) due to its computational efficiency and performance. The third and optional process is referred to as pooling, which again has several advantages. The pooling is a way of summarizing what happens in a region of a feature map, in other words the dimensionality is reduced and information is extracted to a lower dimension. Two pooling strategies worth mentioning are max-pooling and average pooling. Max pooling summarizes the information in a region by taking the maximum value, and the average pooling takes an average of the values in a region. In figure 3.3.11 is a visualization of how the two pooling strategies summarized information in a featuremap. Each 2x2 region in the feature map is transformed into a single value.

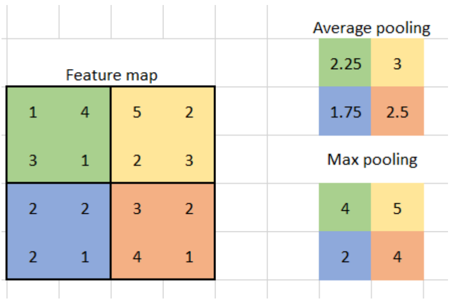


Figure 3.3.11: Average and Max pooling (2x2) on a featuremap

Pooling in general helps to increase translation invariance in latent representations as minor translation produce the same outcome. Local information is lost to create a more global representation of the features of the input. This is great if we value content information over position information.

3.3.2 Learning and Backprogation

As previously mention a CNN propagates information through its convolutional layers by performing convolutions and fully connected by a matrix multiplication between weights and activation. This forward flow of information that produces an output is called forward propagation and it is how the CNN processes information. A CNN as well as other classes of neural networks usually learns useful weights by propagating cost information backwards and use a gradient based method to update the weights such as stochastic gradient decent. The gradient of a specific weight θ with respect to the cost function J, which uses a set of weights θ can be written as

$$\nabla\theta_{i,j,l,c}J(\theta), \tag{3.3.4}$$

where i,j is the index in a kernel identified by c in layer l. To compute the aforementioned expression machine learning libraries usually uses a computational graph such that its possible to backtrack which operations was performed with which inputs to process information. By following this computational graph it's possible to calculate the gradients using the chain rule of calculus. If for example we have an input $x \in \mathbb{R}^m$ and $y \in \mathbb{R}^n$, a function g that maps from $\mathbb{R}^m$ to $\mathbb{R}^n$ and a function f that maps from $\mathbb{R}^n$ to $\mathbb{R}$. Then suppose $y = g(x)$ and $z = f(y)$, then we can calculate the gradient of z with respect to x_i:

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j}\frac{\partial y_j}{\partial x_i} \tag{3.3.5}$$

From this equation it can be seen that the chain rule of calculus is used to compute effect of the weight x_i on the output z through all of the intermediate y-values (y_j refers to the j'th element of the vector y).
In vector notation this can be seen as multiplying the Jacobian of $\frac{\partial y}{\partial x}$ by a gradient $\nabla^z_{\boldsymbol{y}}$:

$$\nabla_{\boldsymbol{x}} z = \left(\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}\right)^\top \nabla_{\boldsymbol{y}} z \tag{3.3.6}$$

This vector notation can be extended to tensor notation, but conceptually it's the same procedure. By recursively using the chain rule, information that is used to update the weights in the CNN is propagated from the last layer to the first. When you have these gradients, they are usually multiplied by a learning rate to form a update rule. This learning rate is one of many important hyperparameters that influence how well a CNN train and perform afterwards.

3.3.3 VGG16

As an introductory investigation of the potential of the VGG16's latent representation of spectrograms, several spectrogram was parsed through the full pre-trained version of the network. The result was that the most commonly predicted label was that of a “theater curtain”. This makes intuitively sense since both theater curtains and spectrograms often contain prominent vertical patterns. A visualization of a spectrogram is shown beneath, and the characteristics that is shared with a theater curtain is comprehensible. Since a spectrogram is processed to a final output of a theater curtain, it seems that the VGG16 is able to capture important distinctive features of a spectrogram. Another general benefit of CNNs' as mention in the section above 3.3.1, is that latent representations are to a large degree invariant to transformations, this makes it possible for a CNN to detect EEG-patterns independently of its position in the spectrogram. Another more speculative benefit is that personal effects may be less represented due to assumption that these effects affect a wide range of spectrum's and there filtered away as background noise. This introductory investigation motivates further investigation the VGG16's latent representation.

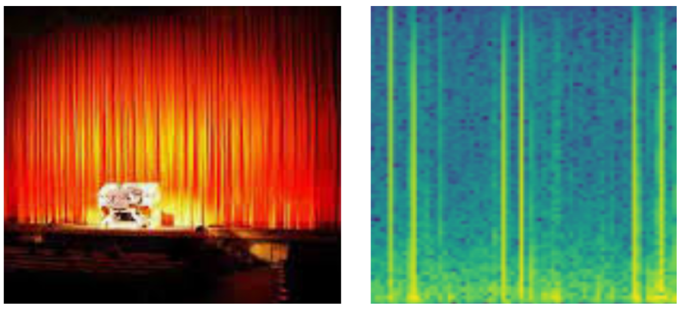


Figure 3.3.12: The image on the left is an image of a theater curtain from the ImageNet data set. The right image is a spectrogram from the BC data set.

Moving on to a description of the VGG16 network. As can be seen beneath in figure 3.3.13 the VGG16 architecture consists of 13 convolutional layers and ends with 3 fully connected layers with a total of 134.268.738 million trainable parameters. The VGG16 network is a deep CNN trained to classify on the well known ImageNet data set with more than fourteen million images of a thousand different classes. It was an architecture that received a top 5 test-accuracy in the 'Large Scale Visual Recognition Challenge 2014'. The ImageNet is often used to benchmark in image classification models, because it requires quite a lot of flexibility of the models to extract useful features to diffentiate between a thousand classes. These features has to be very expressive such that a combination of them can be mapped to the right output. It is this very ability of feature learning that is transferred and taken advantage of when using the pretrained VGG16 to process spectrograms to create a new representation of the EEG signals, that subsequently is used for classification. Because the VGG16 has to be so general to separate the classes, our hope is that it can also create a meaningful latent representation of spectrograms even though it has not been trained on spectrograms.

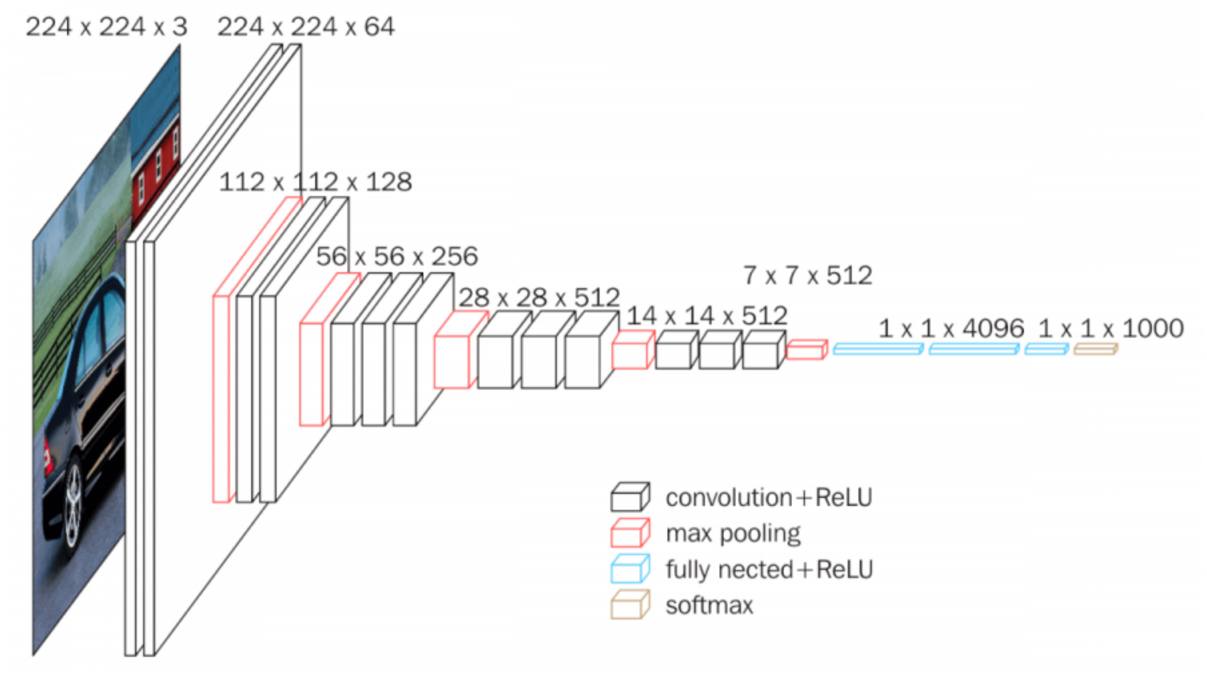


Figure 3.3.13: VGG16 architecture. Visualization taken from [https://neurohive.io/en/popular-networks/vgg16/], 15th of June 2020

3.3.4 How saliency mapping works

Given a input in form of a spectrogram $S_{i,j,k}$ where (i,j) is the pixel index so that $S \in \mathbb{R}^{i \times j}$ and $c = 1, 2, 3$ the colour channel. Given $S_{i,j,k}$ and the target class score K_k where $k = 0, 1$, the goal is to find the gradients $\delta_{i,j,c,k}$ of K_k with respect to the input $S_{i,j,c}$, as shown in equation 3.3.7:

$$\delta_{i,j,c,k} = \frac{\partial K_k}{\partial S_{i,j,c}} \tag{3.3.7}$$

For RGB images this means that each pixel value (i,j) will have a different magnitude for each colour channel k. One way of choosing a single magnitude of the gradient $\delta_{i,j,c}$ for pixel value (i,j) and class c is to take the maximum of the three magnitudes across the channels, see equation 3.3.8.

$$\delta_{i,j,k} = \max_c \frac{\partial K_k}{\partial S_{i,j,c}} \tag{3.3.8}$$

An illustration of saliency mapping using a pretrained VGG16 can be seen in figure 3.3.14. Notice that in this case $k = 1, 2, 3, ..., 1000$ (number of classes in ImageNet).

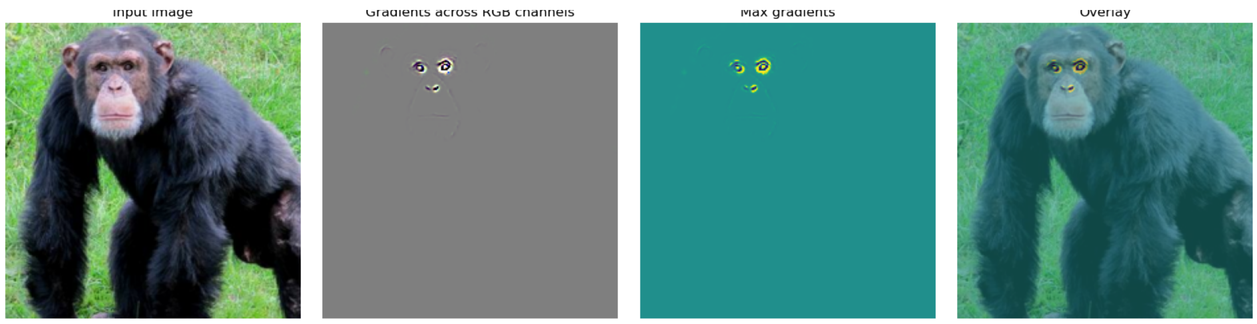


Figure 3.3.14: Saliency mapping of a pretrained VGG16 with gradients across RGB channels and the max gradients

Another view on saliency mapping and measuring gradients of the target class score is that the magnitude of δ determines which pixels needs to change the least to have biggest influence on the target class score C_c [5]. To understand very deep feature representations of images like those in a CNN saliency maps can be used.

3.4 Classification

For classification we are testing the following classifiers: Linear Discriminant Analysis (LDA), Support vector machine (SVM) and Random forest (RF). Explaining the inner workings of the classifiers is beyond the scope of this project, since they will be viewed primarily as a group, and their individual performance will not be discussed. found in [16].

3.5 Evaluation

When evaluating the representations, the prospect of how well they generalize to other training data is ignored. This means that the evaluations should be viewed only in the context of how well the representations perform on the Brain-Capture data used in this study.

3.5.1 K-fold cross-validation

K-fold cross-validation is used when the data set is to small to make a meaning full split into test and training set.

The data is split into folds and each fold is used at a test set while the other folds are used as training set. . For each all classifiers is trained 3.4. And their accuracy can be found as

$$\hat{AC}_m^{\text{gen}} = \sum_{j=1}^{K} \frac{|\mathcal{D}_j|}{|\mathcal{D}|} AC_{\mathcal{M}_s,j}$$

. Where $AC_{m,j}$ is the accuracy on each fold, as a test set. is the classifier and j is the index of the fold, K is the number of folds, $|\mathcal{D}_j|$ is the number of spectrogram windows in the folds test-set and $|\mathcal{D}|$ is spectrogram windows in the entire data set. The reason a weighted accuracy is needed is that we can not be entirely sure that all each fold is exactly the same size. [16].

The fold is satisfied to ensure all classes is represented, even though not usable the lack of data labeled not usable.

3.5.2 Jeffery's intervals

A Bayesian approach can be used to calculate the confidence intervals of the accuracy of a binary classifier. The mathematical formulation being the following

$$p(\theta|m,N) = p(m,N|\theta)p(\theta), \tag{3.5.9}$$

θ being the true accuracy of the classifier, N being the number of samples in the test set and m the number of samples classified correctly. The likelihood of the test results given θ, can be modelled as a sequence of Bernoulli trials

$$p(m,N|\theta) = \theta^m(1-\theta)^{N-m}. \tag{3.5.10}$$

This will be combined with a *Jeffreys prior* which is a non-informative prior distribution for a parameter space. This leads to the following posterior:

$$\begin{aligned} p(\theta|m,N) &= \frac{\Gamma(n+1)}{\Gamma(\frac{1}{2}+m)\Gamma(\frac{1}{2}+N-m)}\theta^{\frac{1}{2}+m}(1-\theta)^{\frac{1}{2}n-m-1} \\ &= \text{Beta}(\theta|a,b), \quad a = m+\frac{1}{2}, \quad b = n-m+\frac{1}{2} \end{aligned} \tag{3.5.11}$$

From this confidence intervals with level of significance $\alpha = 0.05$ will be calculated. For more on the motivation of this approach, and the derivations see [16]. This approach assumes that the samples are independent and identically distributed (i.i.d.).

3.5.3 Comparison of the best classifiers: McNemar's test

McNemar's test is a paired test used to compare two classifiers evaluated on the same test set. It relies on the assumption that what matters when comparing two classifiers is not the observations that are classified correctly or incorrectly by both classifiers, since these could be observations that are trivial to classify or outliers. In a McNemar's test only the observation for which the two classifiers differs in their prediction are of relevance. The hypothesis tested is essentially:

$$r = \frac{n_{12}}{n_{12}+n_{21}} = \frac{1}{2} \tag{3.5.12}$$

With n_{12} referring to the number of times classifier 2 is wrong and classifier 1 is correct and vice versa. Further documentation on the motivation of this method and the calculations of the confidence intervals of performance difference can be found in [16].

3.5.4 Alternative performance estimate of an imbalanced data set

A common problem for most classifications problems is class imbalance. If for example the 'positive' class in a binary classification problem is over-represented a high accuracy could be obtained for a classifier that always predicts a data point to be positive. This however would not be desirable since the classifier have not learned anything about the 'negative' class. In this project a baseline model that always predicts "usable" will achieve a high accuracy of 80 %, why a high accuracy not necessarily reflects the actual ability to seperate the two classes. It is of particular of interest to identify "not_ usable" EEG signals which is the under-represented class because we want to avoid accepting "not_usable" files (false positives) since these signals cannot be analysed by medical personnel in order to detect epileptic discharges. An alternative performance estimate could be the area under the curve (AUC) where the curve refers to the receiver operating charecteristic (ROC) curve. The two key terms are the false positive rate (FPR) and the true positive rate (TPR), shown in equation 3.5.13

$$FPR_\phi = \frac{FP_\phi}{FP_\phi + TN_\phi} \quad TPR_\phi = \frac{TP_\phi}{TP_\phi + FN_\phi} \tag{3.5.13}$$

where FP and TP is normalized with the size of each class. The size of the 'negative' class will be the number of false positives (FP) plus the number of true negatives (TN). The size of the 'positive' class will be the number of true positives (TP) plus the number of false negatives (FN). If FPR and TPR are plotted for different thresholds ϕ the ROC curve is formed and the AUC can be calculated. Because of the normalization this accuracy measure will be independent of the prior odds for the classes. Documentation can be found in [16].

3.6 Visualization

When making a dimensional reduction, it is important to note that the BC data-set is a so call ill-posed data-set because it has more dimensions than data points, which would make the reduced space vulnerable to outliers [1]. The fact that so much of the data set has one label would probably make a bias to-wads finding a representation for clustering that label best. Therefore dimensional reduction was made so as many files as possible was used while still ensuring balance in the labels.

3.6.1 Principal Component Analysis (PCA)

PCA is used to find the directions in a data-set that carries the most variation. It is done by making a singular value decomposition(SVD) [1]. The magnitude of the eigenvalues determines the amount of variation that the corresponding eigenvectors represent. The first two principal components can be used to project all data onto the 2D-plane with the largest variation. PCA is effective at determining large-scale trends in a data-set.

3.6.2 t-distributed Stochastic Neighbourhood Embedding (t-SNE)

t-SNE is a method for visualizing high-dimensional data by calculating conditional probabilities of points based on euclidean distance in the high dimensional space, and modelling these probabilities in a lower dimensionality. The conditional probabilities $p_{i|j}$ in the high-dimensional space are calculated using Gaussian functions

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|/2\sigma_i^2)}{\sum_{i \neq j} \exp(-\|x_i - x_k\|/2\sigma_i^2)}, \tag{3.6.14}$$

σ_i^2 being the variance of the Gaussian centered in x_i. For documentation on the methods for determining σ_i^2 see [2]. In a lower dimensionality these joint probabilities are modelled using a heavy-tailed Student t-distribution with one degree of freedom, allowing dissimilar objects to be mapped further away. The conditional probabilities of the low dimensionality are optimised according to the Kullback–Leibler divergence, which is a measure of the difference between two probability distributions. The advantage of t-SNE over PCA is that it catches more local structures in the data, and it is not confined to linearity.

3.7 Unsupervised clustering

To train a GMM the Expectation Maximization (EM) algorithm is utilized to to estimate the parameters $\vartheta : \{\mu,\mathbf{\Sigma},\pi\}$. The EM algorithm maximizes the log-likelihood of the data where a regularization parameter λ is added to $\mathbf{\Sigma}_g$ in the EM algorithm. A GMM with G Gaussian mixture components (densities) can be defined as in equation 3.7.15. The GMM and corresponding training algorithm is based on the documentation found in [16].

$$p(\tilde{\mathbf{x}}) = \sum_{g=1}^{G} p(\tilde{\mathbf{x}}|\vartheta_g) \cdot \pi_g \tag{3.7.15}$$

4 Method

4.1 Overview of experimental setup in relation to research questions

We will utilize the binary 'is-usable'-label stating whether an EEG-recording is usable for clinical trials, to test two different data representations of the EEG-recordings from the BC data set. The first data representation is spectrograms generated from EEG-signals. The second is a pretrained VGG16's latent feature representation of the same spectrograms. To investigate research question 1 classification accuracies for different machine learning models will be compared to a baseline model. The comparison will be performed by first looking at Jeffreys confidence intervals, then a paired McNemar test and finally ROC curves with AUC.
For research question 2 the same results will be analysed together with confusion matrices to evaluate which data representation that best suits the classification task. An initial visualization study of the two data representations with PCA and t-SNE will serve as a qualitative tool for explaining the results we obtain.
To investigate research question 3 concerning explainability of the CNN's perception of a EEG-spectrogram saliency mapping of selected 'not_usable' spectrograms will be analysed and compared to corresponding raw EEG-signals. Saliency mapping will further be used to examine the quality of the feature extraction as sanity check of whether transfer learning can be utilized on the BC data set investigated in research question 2.
At last, we aim to investigate research question 4 regarding unsupervised clustering of EEG-windows labelled "not_usable" by fitting a Gaussian Mixture Model to the feature vectors and visualize multiple raw EEG-signals within each cluster to potentially identifying certain characteristics.

4.2 Preprocessing

4.2.1 Slicing EEG-recordings

The EEG recordings were separated into windows of 120 seconds for each of the 14 channels for each of the 126 files. When sliding over the EEG-signal the window is only moved 60 seconds at a time, to ensure that potential artefacts of interest located at the boundaries of the window, are properly represented in at least of the windows containing all or parts of it. This yields a total of approximately 150.000 windows.

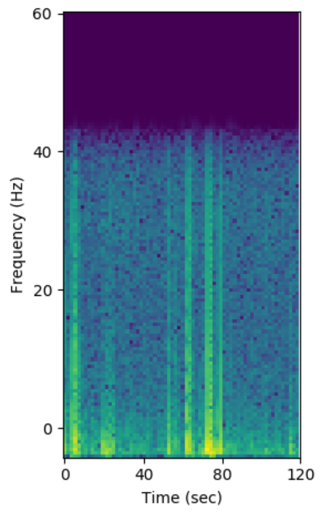
4.2.2 Filtering

All raw EEG data was processed by a band-pass filter, rejecting all signals below 1 Hz and higher than 40 Hz This was done because brainwaves are primarily found in this spectrum [4]. Afterwards a line-noise notch filter in the 50 Hz area was used to eliminate noise from the electric grid. This is a standard procedure when processing EEG [6]. Furthermore the average of the channels at the n'th timestep is used as a reference and subtracted from each channel. This is done to diminish the effect of non-neural sources in the signal. Usually a reference channel, positioned close to the other electrodes but shielded from neural activity, is used.

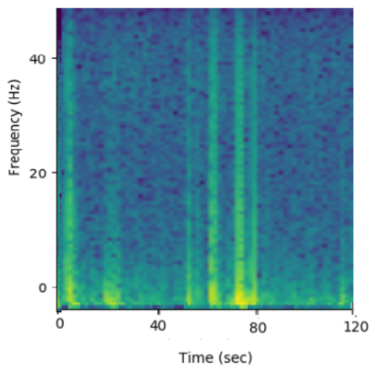
4.2.3 Spectrograms

The spectrograms were generated using Fast Fourier Transform on the EEG-windows, with time-segments of 1.75 seconds and frequency-steps of 0.5 yielding 68 time-steps and 128 frequencies between 0 and 64. 50% overlapping, between time-segments is used for smoothening the signal, and to avoid the spectrograms being overly sensitive to the starting and end point of each frequency time-segment. The machine learning models trained directly on spectrograms, are given the concatenated log intensities of the frequencies contained in each of the 14 channels. The flattened vector yields a vector of size 122.000.

For the spectograms to be passed through the VGG16 network, the spectograms are rendered to RGB-format and resized such that the input dimensions become 3x224x224. Due to the limited format accepted by the CNN frequencies above 45 Hz are cropped, before resizing the picture, to ensure that only the most valuable information is kept. The pictures are then normalised according to the standard deviation and mean of the ImageNet data set, since this is the input that the VGG16 is optimised to handle [25]. The two representations of the spectograms can be viewed in Figure 4.2.15.



(a) Spectrograms passed directly to ML models



(b) Resized and cropped spectrograms to be passed through CNN

Figure 4.2.15: The two types of spectrograms passed directly to ML models (right) or processed through the CNN (left)

4.3 Labeling the data

Since annotations are not made directly on non usable passages in the EEG-recordings, some assumptions had to be made when labeling the spectrograms. It was assumed that in a 120 second time-window EEG-recordings labeled "not_usable" will show signs of the disturbances that makes them unusable, and that these disturbances will not appear in usable EEG-recordings. This leads to labeling the spectrograms according to the label of the recording that they originate from. This is a very naive assumption, that will most likely not hold for all windows, since it is to be expected that recordings labeled not-usable have some time periods with normal EEG and vice versa. However if this is true for a large proportion of the windows a ML-model can learn to outperform a baseline. This is the motivation for research hypothesis 1.

4.4 Evaluation

The two data representations were tested against a baseline which guesses on the largest class ("usable"). The machine learning models used for classification are Random Forest, Support Vector Machine, and Linear Discriminant Analysis. The two latter models are described in section 3.4, and knowledge of RF is assumed. All classifiers are imported from Sci-kit-learn. The classification accuracy was measured based on classification of each individual time-window.

4.4.1 Hyperparameter tuning

We have decided to include a large variation of models with the downside of letting the hyperparameter tuning be a task for further research. With the amount of high-dimensional data and hyperparameter combinations, it was simply not feasible to do a proper hyperparameter tuning. The models were therefore initialised with default parameters from Sci-kit-learn.

4.4.2 Data splits

Two experiments were carried out with the only difference being the training sets. Both were performed using a five fold cross-validation. For both training sets the cross-validation splits were made on recording-level to avoid windows from the same recording appearing in both test and training set. This yielded approximately 26 recording in each test split and 100 recording in each training split. The splits were stratified in order to make the number of recordings from each class reflect the true priors of the classes.

The first experiment was performed with a training set reflecting the true priors of the two classes. To diminish computations, only around 20 windows from each file were included. The windows were picked with an equal amount of time between the windows based on the length of the recordings to best reflect the whole recording.

The second experiment was performed on a training set with an even amount of windows from each class. The training set was balanced by including more windows from the "usable" class from each recording, while keeping the total amount of windows in the training data approximately the same as in the other experiment. This yields around 13 windows for each recording of the "usable class and 50 windows from each class of the "not_usable" class.

An overview of the distribution of windows and files in the two training sets and test sets can be found in table 4,5, 6.

Fold	1	2	3	4	5
N_files, "usable"	80	80	80	80	80
N_files, "not_usable"	20	21	21	21	21
N_windows, "usable"	1618	1614	1609	1624	1615
N_windows, "not_usable"	394	416	426	410	410

Table 4: Imbalanced training set with true priors on the classes for first experiment

Fold	1	2	3	4	5
N_files, "usable"	80	80	80	80	80
N_files, "not_usable"	20	21	21	21	21
N_windows, "usable"	982	982	984	984	980
N_windows, "not_usable"	1040	1042	1099	1040	1070

Table 5: Balanced training set for second experiment

Fold	1	2	3	4	5
N_files, "usable"	20	20	20	20	20
N_files, "not_usable"	6	5	5	5	5
N_windows, "usable"	803	819	777	794	795
N_windows, "not_usable"	208	208	176	207	198

Table 6: Test set used for both experiments

4.4.3 Statistical analysis

The test results from each fold are concatenated and the confidence interval for the accuracy of each representation, is calculated using the Jeffreys intervals for both experiments. The best classifiers for each representation for each experiment are then compared using a McNemar's test, on the predictions from each classifier. To get an alternative estimate of how the classifiers, trained on a balanced and imbalanced data set respectively, performed the ROC curve was plotted and the AUC calculated. Confusion matrices were also made for all classifier on each representations. The validity of the hypotheses are evaluated based on an overall impression of these measures.

4.5 Transfer learning with VGG16

4.5.1 Generating feature vectors

Initially the pretrained VGG16 was used for generating feature vectors. This was done by parsing spectrograms of size $3 \times 224 \times 224$ (each spectrogram representing 1 of 14 channels for a specific window of a specific file) through the CNN and afterward concatenate the latent feature vectors each of size 1×4096 (see figure 4.5.16). The feature vector for one window will then be $14 \times 4096 = 57.344$ dimensional, less than half of the original vector containing concatenated spectrograms for one window of size 122.000. The VGG16 latent feature vectors were cut out as late as possible, just before the fully connected layer representing the 1000 classes, the CNN was trained to distinguish between. This was chosen to keep as much of the pretrained VGG16 feature representation as possible. Other possibilities within the classifier part of the VGG16 was from the $7 \times 7 \times 512$ max-pooling layer or the first of two dense layers of dimensions 1×4096. Both the spectrogram vector and the generated feature vectors will be passed to several machine learning models to evaluate the two data representations.

4.5.2 Tuning the pretrained VGG16

To see if VGG16 could learn a better latent representation of the EEG signals, training, with the goal of classifying the binary label "usable" or "not_usable", was carried out. The use of transfer learning, using the pretrained VGG16 in order to obtain an improved feature representation of the EEG-signals, was justified with the argument that the BC data set is far too small to train a deep CNN from scratch. To adjust the VGG16 to the binary classification problem the the classifier part (the fully connected layers layers, see 4.5.16 CNN1) of VGG16 were modified while the input dimensions remained the same. It is important to note that the CNN is trained on the basis of a single channel, whereas the machine learning models (RF, SVM and LDA) uses information from all 14 channels. As shown in figure 4.5.16 there is a fully connected layer from the latent feature vector (size 1×4096) to a vector of size 1000 (ImageNet classes). The altered CNN will have a output vector of size 2 for the two classes ("usable" / "not_usable").

To control the number of layers tuned, we froze all but the layers of interest in terms of tuning. To investigate if a better latent representation can be obtained two different CNN's were tested on the BC data set:

- CNN1: Only the classifier are trained
- CNN2: The $14 \times 14 \times 512$ convolutional layers are trained together with the classifier

Training more convolutional layers was unfeasible with the limited amount of data. Additionally only one data split was evaluated. An overview of where in the VGG16 the latent feature vector is placed and the trainable parameters for CNN1 and CNN2 is shown in figure 4.5.16.

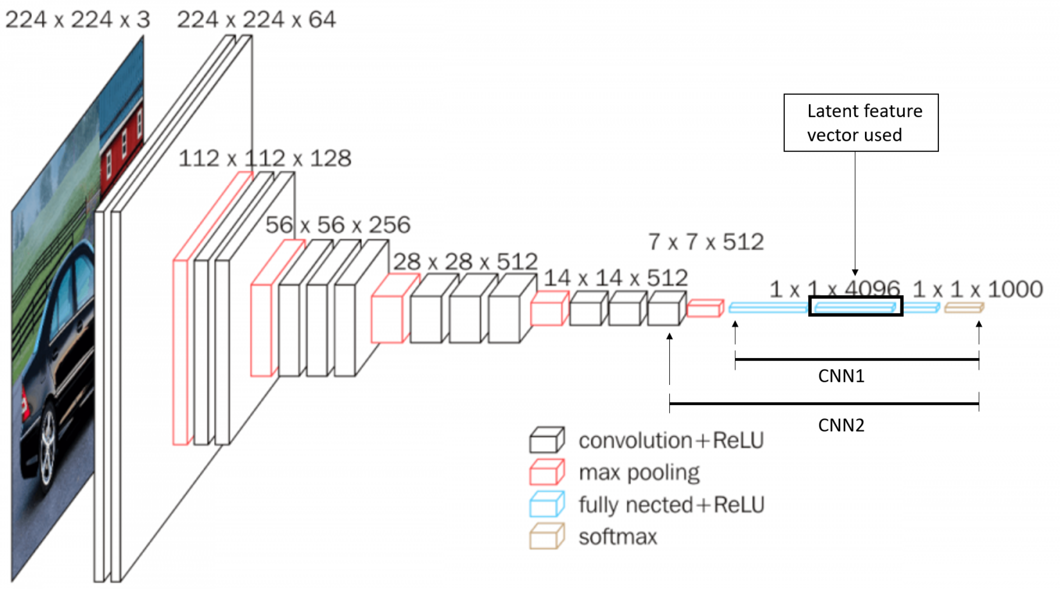


Figure 4.5.16: Illustration of the latent feature vector used from VGG16 and the trainable parameters for CNN1 and CNN2

4.5.3 Experiments with the CNN

Because of the imbalance in data set and the naive assumption that labeling is consistent to each time-sigment within a file, both CNN1 and CNN2 mainly predicted "usable" on the test set in a separate pilot study. In order to avoid this tendency and handling the imbalance the two CNN's were also trained on a balanced training set as described in section 4.4.2. The batch size is 256, number of epochs 2 and Adam optimizer with different learning rates = [0.0001, 0.0005, 0.001, 0.01, 0.02] for both CNN1 and CNN2 trained on the balanced or imbalanced training set. The CNNs all use a cross-entropy loss function and the maximum number of windows from each file is 20 to reduce computation time. The 20 windows chosen was evenly spread out over the recording (and thereby account for 40 minutes) and all 14 channels were included.

4.6 Explainability of CNN with saliency maps

The use case of saliency mapping in this project was to investigate which areas in the spectrograms a selected CNN focuses on when deciding whether or not the time-segment of the EEG signal is usable for clinical purposes or not.

The same CNN chosen for generating feature vectors for clustering was used to perform saliency mapping. Saliency mapping was utilized using a Python visualization toolkit for neural network named FlashTorch, see blogpost by creator in [28]. For the abnormal data set a number of spectrograms was selected and passed to the retrained CNN to generate saliency maps. The corresponding raw EEG signals was plotted to further analyze the spectrograms. These saliency maps served as explainability of the CNN's perception of EEG signals through deep feature representations.

4.7 Clustering in latent space

The best performing CNN with respect to correctly classified "not_usable" EEG signals (based on most correct prediction on this class) was selected to generate new feature vectors. The justification of using new feature vectors instead of the spectrogram representation was that the tuned feature layers in the CNN seemed to improve its ability to correctly predict "not_usable" single-channel spectrograms. Additionally the latent representation from a CNN is more invariant to simple feature transformations like shifting and scaling as mentioned in 3.2.2, which is not the case for the spectrogram representation. This new data set of correctly predicted "not_usable" spectrograms in the test set will be refered to as the abnormal data set from now on. The features of a EEG signal will more likely be normally distributed if an appropriate feature extraction method is used [3]. The reason for choosing GMM as a clustering method are the promising prospects in detecting anomalies in EEG signals shown in [15] (see section [1.1] for more details).

To reduce the dimensions of the high-dimensional latent feature vectors while maintaining most of the data variability PCA was performed. For this project, a GMM was fitted to the 1039 dimensional (number of correctly predicted "not_usable" windows) principal components of the latent feature vector β, since the goal is to derive a distribution for the vector β. The GMM was evaluated for several different numbers of centroids, and was visualized in a 2D PCA plot to check if potentially meaningful clusters was found in the PCA space of lower dimensionality. For this reason the number of centroids chosen to move forward with was four. Afterwards the clusters found by the GMM, were visually evaluated based on observations chosen from a 2D PCA plot. A description of how to interpret EEG signals and characteristics of artifacts can be found in section 2.2.

4.8 Visualization

In order to visually inspect the two representations , 30 windows of 10 different recordings, 5 of each label, were plotted in 2D according using the two dimensions reduction described in section 3.6. These were plotted in relation to their their binary label, and to the recording they originated from In order to reduce bias in the dimension reduction we fitted the PCA and TSNE on 52 files 26 of each label. To account for the possibility that the variation between windows based on the binary 'usable/not-usable' label might be confounded by the variation between windows from the same recording, plots of the same windows for both techniques was made for comparison. Visualiation of some of the time-windows of the clusters of usable and non-usable EEG will be made, to investigate if these windows are in fact labeled wrong, according to the naive labeling described in section 4.3.

4.8.1 Visualization as traceable

Being able to trace back to the original EEG signal and spectrograms is importent to the project in order to get an intuitive understanding of the reduced space and how artefact may cluster together. For further study external expert without coding experience may need to label some of the artefact found. Therefore a great deal of coding effort was put into making interactive visualisation plots with the ability to trace the plot back from the reduced space to the spectrogram and the original EEG data and get meta data by just hovering the mouse above the plot. This can also be utilized to understand type errors made by the classifiers. The EEG data visualisation support a zoom function that would make it easy to see specific artefact in those windows. Plots made with this function can be found in appendix 8.7.

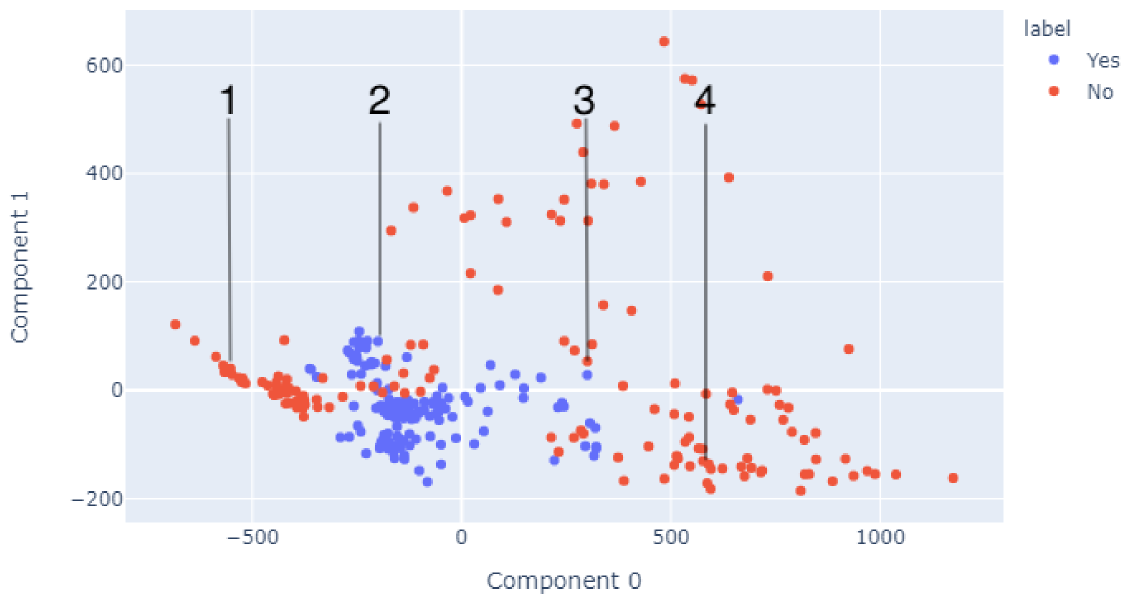
5 Results and analysis

5.1 Visualization

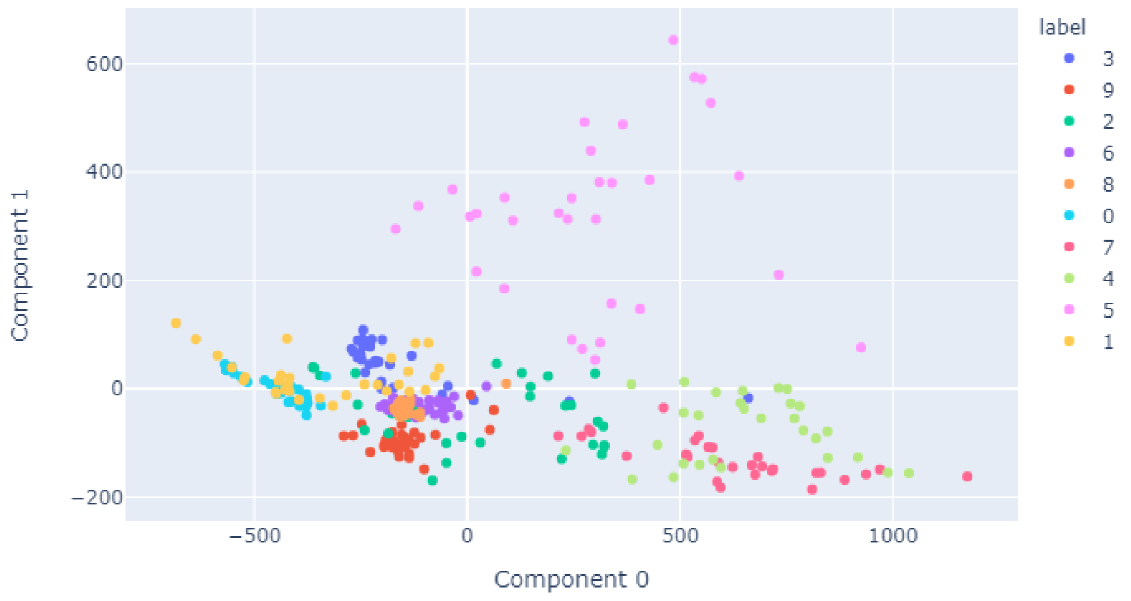
In figure 5.1.17, the spectrogram representation is visualised using PCA. It is clear to see from the second figure 5.1.17b that the clusters in figure 5.1.17a are mainly due to similarities between windows from the same recordings, and not the label stating whether the observation is fit for clinical purposes, which could be the impression by only looking at figure 5.1.17a. However it does seem like multiple recording-clusters with the same label are clustered together or adjacently. It also seems like the windows from the recordings that are labeled "not_usable" are clustered closer together than the windows from recordings that are labeled "usable", however this can only be confirmed on the basis of the first two principal components.

In figure 5.1.18, the first four spectrograms of the EEG-channel "TP-10" (located at the middle, far right) shows the numbered observations in figure 5.1.18a. Only one channel for each window is included in the report for simplicity.

When visually comparing the spectrogram in figure 5.1.18a from the "not_usable" cluster with the spectogram figure 5.1.18b "is-usable" cluster, there is no apparent difference, implicating that the assumption in section 4.3 is problematic. From the PCA-plot figure 5.1.17a observation 3 seems to be an outlier of the is-usable class, which is confirmed by looking at the corresponding spectrogram figure 5.1.18c 3, with a spike around the 30 seconds, which is also present when inspecting the other channels of the same window. Spectrogram figure 5.1.18d 4 is in the middle of a cluster of windows labeled not-usable, and seems to consist of many artefacts disturbing the signal, correctly reflecting the label not-usable. In general from the visual inspection of the spectrograms of the observations it seems that observations located far away from the majority of the observations contains unwanted artefacts. However some usable recordings have windows that are located far away from other observations, and some not-usable recordings have windows that are only located close to the majority of the observations.

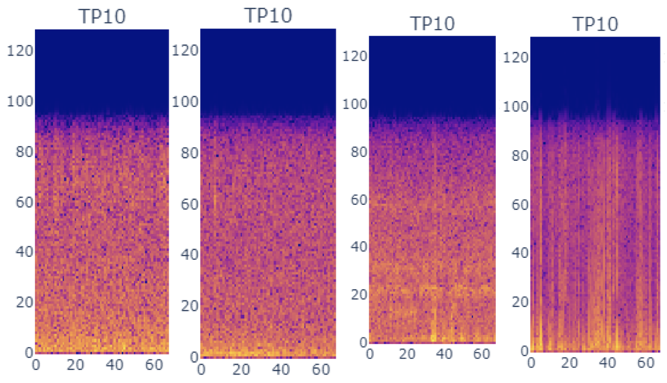


(a) PCA on spectrograms with labels referring to whether the EEG-window is useable



(b) PCA on spectrograms with labels referring to a specific recording

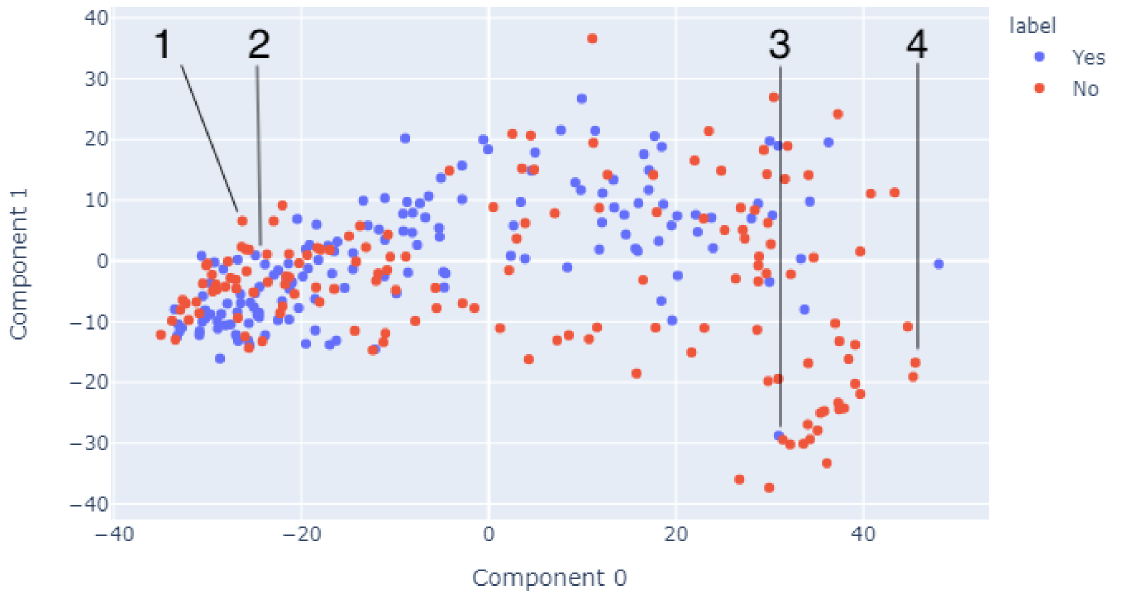
Figure 5.1.17: The two first principal components on spectrogram representation. The observations are the same in the four plots. **The numbered observation in plots (a) and (b), are visualised as spectrograms in figure 5.1.18**



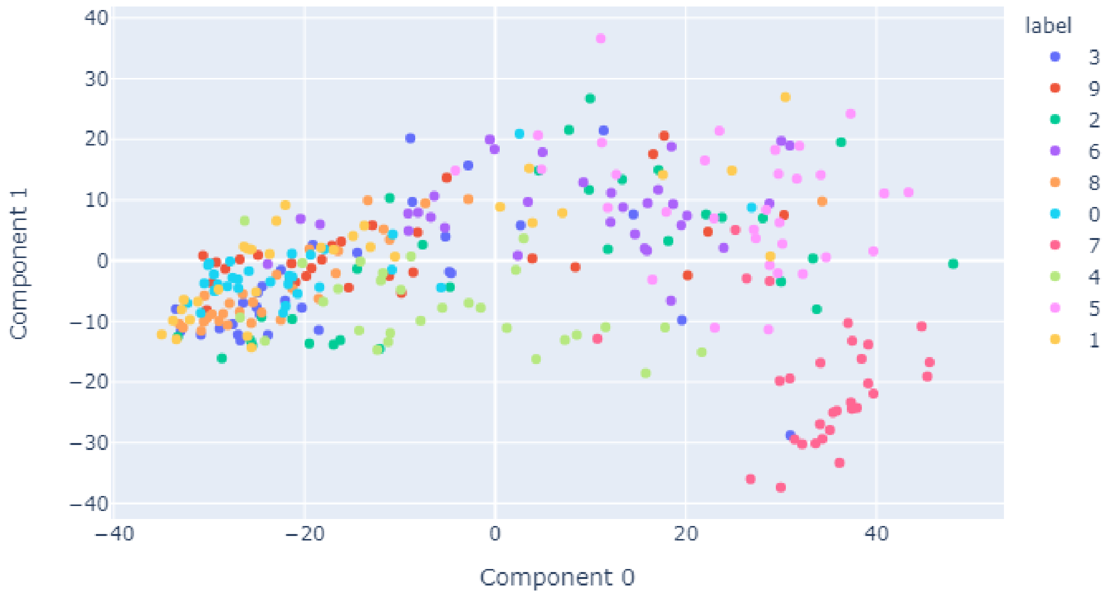
(a) 1, "not-usable" (b) 2, "is-usable" (c) 3, "is-usable" (d) 4, "not-usable"

Figure 5.1.18: Spectrograms of numbered observations in figure 5.1.17. Here the x-axis being the number of timesteps and the y-axis the frequency in Hz.

In figure 5.1.19 the feature vector representation is visualized using PCA. From looking at the first figure 5.1.19a it seems that the classes are not as well separated as for the spectrogram representation, although from looking at the second figure 5.1.19b, it seems that this is mainly due to that fact that windows from the same recordings are not clustered together. This shows that the feature vector representation is not as dominated by the group structures of the recordings as we saw for the spectrograms. However it cannot be excluded that they would be visible for other principal components. From figure 5.1.20 the numbered observations in figure 5.1.19 are visualized. It seems that observations that are far away from the majority of the observations (5.1.20c 3, 5.1.20d 4) are more prone to contain artefacts, with the opposite being true for observations located close to the majority of observations (5.1.20a 1, 5.1.20b 2). It seems that not-usable windows are marginally over-represented in being located far away from the majority of the observations, but it is not enough to seem significant.

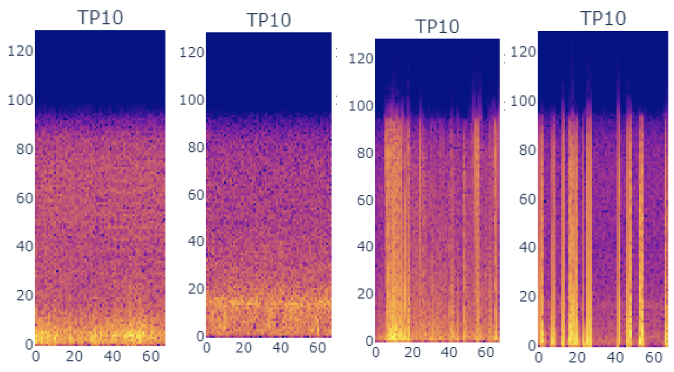


(a) PCA on feature vectors with labels referring to a specific recording



(b) PCA on feature vectors with labels referring to a specific recording

Figure 5.1.19: The two first principal components on feature vector spectrograms. The observations are the same in the four plots. **The numbered observation in plots (a) and (b), are visualised as spectograms in figure 5.1.20**

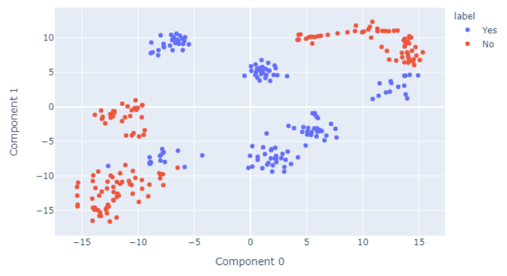


(a) 1, "not-usable" (b) 2, "is-usable" (c) 3, "is-usable" (d) 4, "not-usable"

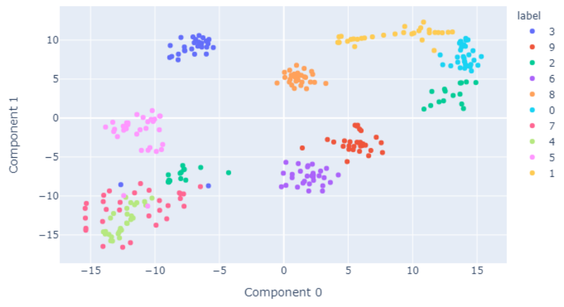
Figure 5.1.20: Spectrograms of numbered observations in figure 5.1.19. Here the x-axis being the number of timesteps and the y-axis the frequency in Hz.

The t-SNE of the spectrogram shown in figure 5.1.21, also clusters windows from the same recording closest together. It also seems that there might be some tendency that the clusters of the windows of the "usable" class are more dense. This might however be an effect obtained by projecting data onto a 2D-plane.

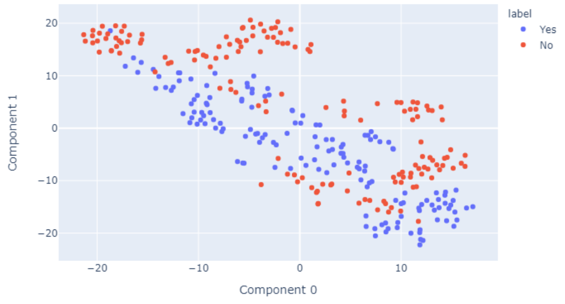
The t-SNE of the feature vectors shown in figure 5.1.21, also reflect the clustering of windows from the same recording, but it is not as significant as for the spectrogram representation. It does not seem like there is any tendency that the clusters of the windows from recordings with the same level are closer together.



(a) t-SNE on spectrograms with binary labels

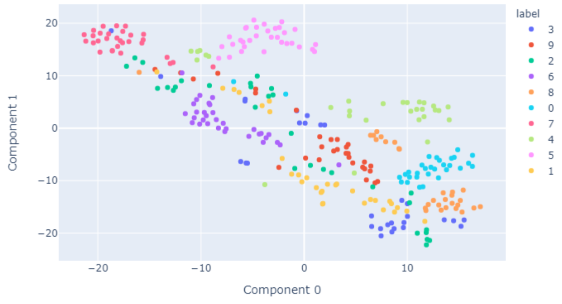


(b) t-SNE on spectrograms with labels referring to the specific recording



5

(c) t-SNE on feature vectors with binary labels



(d) t-SNE on feature vectors with labels referring to the specific recording

Figure 5.1.21: The two first t-SNE components on feature vectors and spectrograms. The observations are the same in the four plots and in figure 5.1.17 and 5.1.19

The main implication of the visualization are:

- The overlap between the classes are bigger for the feature vector representation than for the spectrogram representation.
- Windows from the same recording seems to be clustered together for spectrograms, but this effect is not as strong for feature vectors.
- The further away an observation is from the majority of the observations the more likely it is to contain what seems to be unwanted artefacts.
- Clusters of not usable EEG does not seem to necessarily contain unwanted disturbances.

5.2 Accuracies and Jeffreys intervals

The accuracies and Jeffrey confidence intervals of the classifiers, when classifying windows for their clinical usability, can be found in the following tables 7 and 9. The results in 7 shows the accuracy of the ML-classifiers on the imbalanced training and test set. It can be seen that the spectrogram representation performs statistically better than the baseline for all three models, and the feature representation significantly better for RF and LDA. These results provide support for hypothesis 1, which says that it is possible to construct a classifier that is superior to a baseline model.

RF and SVM perform significantly better on the spectrogram representation compared to the feature vector representation. The best classifiers on the spectrogram representation (RF) perform significantly better than the best models on the feature vector representation (RF & LDA). These results contradict hypothesis 2, which states that the latent feature vector representation is superior to the spectrogram representation.

When looking at the accuracy over the folds in table 8 the variation is very large for both representations. This is an indication of the group structures of the recordings, which is also evident from the visualizations in section 5.1.

Model	Accuracy (F)	CI (F)	Accuracy (S)	CI (S)
RF	0.82	[0.81, 0.83]	0.86	[0.85, 0.87]
SVM	0.81	[0.80, 0.82]	0.85	[0.84, 0.86]
LDA	0.82	[0.81, 0.83]	0.83	[0.82, 0.85]

Table 7: Classifiers generalisation accuracy's on a naturally imbalanced data set. The training data was either spectrograms (S) or feature vectors (F).

	Accuracy (F)					Accuracy (S)				
Model\Fold	1	2	3	4	5	1	2	3	4	5
RF	0.83	0.79	0.83	0.82	0.80	0.90	0.81	0.88	0.84	0.87
SVM	0.86	0.80	0.80	0.81	0.78	0.91	0.88	0.84	0.76	0.86
LDA	0.86	0.80	0.78	0.81	0.80	0.88	0.85	0.83	0.76	0.84

Table 8: Accuracies on each of the 5 folds with natural priors in training set. Mean accuracy and confidence intervals are shown in table 7.

Table 9 shows the accuracy of the ML-classifiers trained on a balanced training set, and test set with natural priors. All classifiers trained on the feature vector representation performs statistically worse than a baseline. However two classifiers (SVM & LDA) still perform significantly better than a baseline on spectrograms, supporting hypothesis 1. All classifiers trained on the spectrogram representation outperforms the classifiers trained on the feature vector representation significantly, contradicting hypothesis 2. When looking at the accuracy over the folds in table 10 the variation over the folds is again very large, with the biggest difference in accuracy between folds for the same model on the same data representation being more than 20 % in some cases.

Model	Accuracy (F)	CI (F)	Accuracy (S)	CI (S)
RF	0.72	[0.70, 0.73]	0.81	[0.80, 0.82]
SVM	0.77	[0.76, 0.78]	0.84	[0.83, 0.85]
LDA	0.76	[0.75, 0.77]	0.82	[0.81, 0.83]

Table 9: Classifiers generalisation accuracy's on a balanced training set and naturally imbalanced test set. The training data was either spectrograms (S) or feature vectors (F).

	Accuracy (F)					Accuracy (S)				
Model\Fold	1	2	3	4	5	1	2	3	4	5
RF	0.79	0.77	0.57	0.72	0.70	0.91	0.82	0.71	0.84	0.78
SVM	0.82	0.79	0.70	0.74	0.77	0.91	0.87	0.78	0.76	0.86
LDA	0.84	0.80	0.63	0.76	0.75	0.88	0.85	0.83	0.76	0.83

Table 10: Accuracies on each of the 5 folds with a balanced training set. Mean accuracy and confidence intervals are shown in 9.

5.2.1 McNemar's test

Two 3x3 paired McNemar tables are shown in table 11 and table 12. These tables contain the results of the paired McNemar test performed on the two best classifiers for each representation and the baseline model. The best performing classifier on an imbalanced data set according to table 7 is a RF classifier, and the SVM classifier in table 9 with a balanced training set.

From table 11 it can be seen that the baseline model statistically is outperformed by both the RF on spectrograms and the RF on feature vectors slightly. Furthermore it can be seen that the spectrogram representation makes it easier for an RF classifier to separate the two classes. This is also what was implicated in the visualization, see section 3.6.

	Baseline	RF (F)	RF (S)
Baseline	▨	-0.02 [-0.02:-0.01]	-0.06 [-0.6:-0.05]
RF (F)	0.02 [0.01:0.02]	▨	-0.04 [-0.04 :-0.03]
RF (S)	0.06 [0.05:0.06]	0.04 [0.03 :0.04]	▨

Table 11: Pairwise model comparison on a naturally imbalanced training and test set. The first value is the difference in model accuracies $\theta_{row} - \theta_{column}$ and the second is the confidence interval of this measure.

From the table 12 it can be seen that only the SVM on the spectrograms statistically outperforms the baseline model, whereas the SVM on feature vectors has worse performance. As for the RF, the SVM also classifies better on the spectrogram representation.

	Baseline	SVM (F)	SVM (S)
Baseline	▨	0.03 [0.02:0.04]	-0.04 [-0.5:-0.03]
SVM (F)	-0.03 [-0.02:-0.04]	▨	-0.07 [-0.08 :-0.06]
SVM (S)	0.04 [0.03:0.05]	0.07 [0.06 :0.08]	▨

Table 12: Pairwise model comparison on a balanced training and naturally imbalanced test set. The first value is the difference in model accuracies $\theta_{row} - \theta_{column}$ and the second is the confidence interval of this measure.

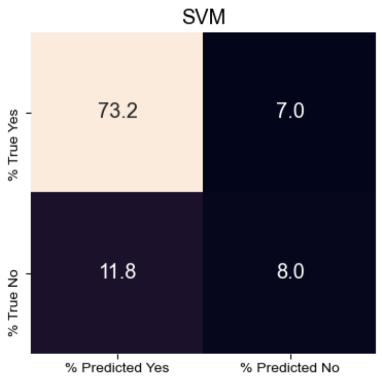
These results show the same as section 5.2, supporting hypothesis 1 but contradicting hypothesis 2. However caution has to be taken since the McNemar's test is also sensitive to group structures.

5.3 Confusion matrix

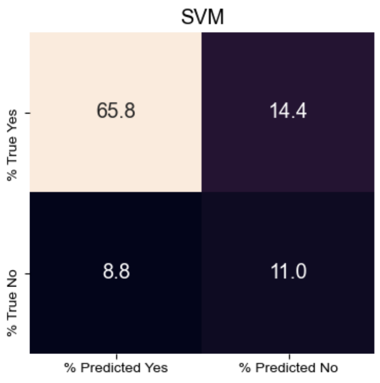
The two following sections will focus on the performance between representations on the two different training data sets. For convenience only the results of the SVM on feature vectors and the RF for spectrograms will be included in the report. The rest can be found in appendix 8.3.

The confusion matrices in figure 5.3.22 show that for the balanced data set, the best classifier on the spectrogram representation RF has a better overall accuracy than the best model on feature vectors SVM, because it is better at predicting the windows that are labeled "usable". However when trained on a balanced training set the classifiers on the spectrogram representation obtains a better accuracy because it is better at predicting windows labeled "not_usable". Although when looking at the confusion matrices in appendix 8.3 for the other classifiers this is not a general trend for the differences between classifiers trained on spectrograms and feature vectors, when comparing the performance on an unbalanced training set and a balanced training set. The larger performance gap between classifiers trained on spectrograms compared to classifiers trained on feature vectors, when shifting from an imbalanced data set to a balanced data set, seem to be because the feature vector classifiers are more prone to overfitting to the prior of the classes.

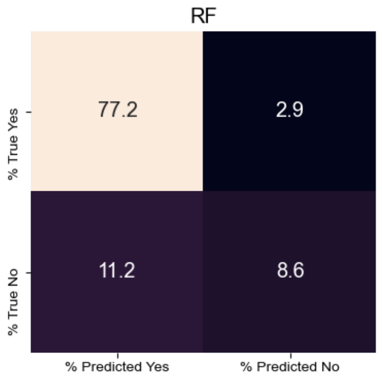
This again supports hypothesis 2, that the spectrogram representation is better than the feature representation.



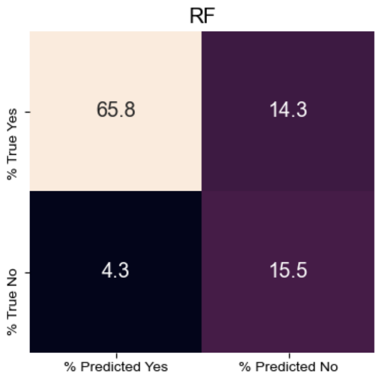
(a) Unbalanced feature vectors



(b) Balanced feature vectors



(c) Unbalanced Spectrograms



(d) Balanced Spectrograms

Figure 5.3.22: Confusion matrix for the best classifier for each experiment. All confusion matrices can be found in appendix 8.3

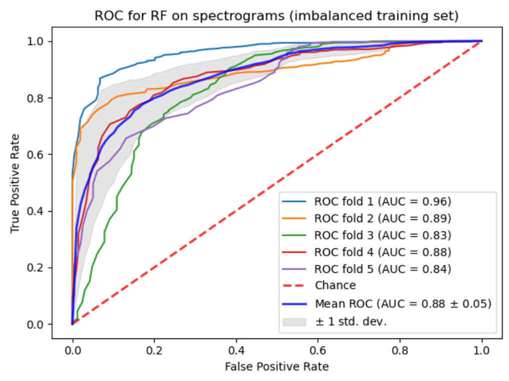
5.4 ROC curves and AUC

From figure 5.4.23 the ROC curves for the best performing classifiers for each representation can be seen, the rest can be found in appendix 8.4. It does not seem like there is any significant difference on the AUC, between using a balanced or an imbalanced training set. This seems to be the case for all classifiers on both representations. The same variation over the folds is visible for this measure as seen in section 5.2 for accuracy.

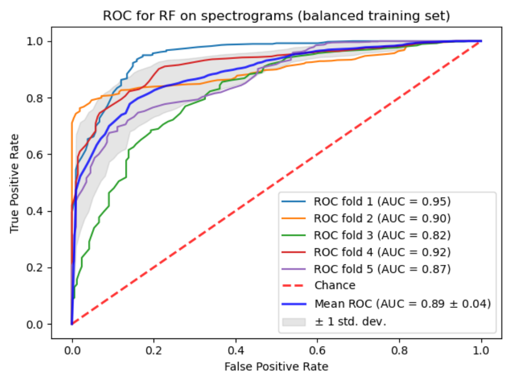
For the spectrogram representation RF has the highest AUC for both training data sets with 0.88 on the imbalanced and 0.89 on the balanced. RF also has the lowest variation in performance over the folds for both data sets. However the difference in AUC between random forest and the other classifiers trained on spectrograms does not seem to be significant.

For the feature vector representation the SVM has the highest AUC with 0.79 together with RF for the imbalanced data set. For the balanced data set SVM also has the highest accuracy together with LDA of 0.79. The variation between the folds is lowest for the SVM.

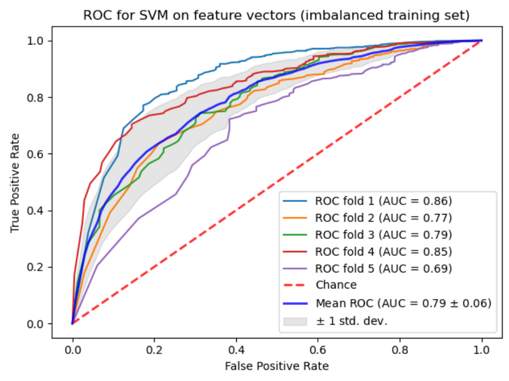
When comparing the two representations, classification on spectrograms seems to outperform classification on feature vectors based on their AUC. This is true when comparing the best classifiers for each representation since RF on spectrograms outperforms SVM on feature vectors in all folds. The same is true when comparing each classifiers trained on both representations since all classifiers seem to perform better when trained on spectrograms. From this measure it is even more clear that the separation between the classes is much better for the classifiers trained on spectrograms compared to feature vectors, which was also apparent in the visualization. RF on spectrograms is able to obtain a TPR of close to 0.5, with a FPR close to zero on a balanced set, where the SVM obtains a TPR of less than 0.1 for the same conditions. This further disproves hypothesis 2.



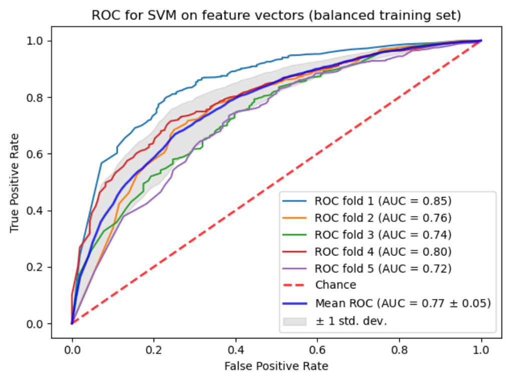
(a)



(b)



(c)



(d)

Figure 5.4.23: ROC and AUC for RF and SVM on feature vectors and spectrograms

5.5 Results for CNN's

The accuracies and Jeffreys confidence intervals of the two CNN's trained on a balanced and imbalanced training set are shown in table 13 and 14. In general both types of CNN achieves higher accuracy when trained on the imbalanced training set, with CNN1 being the superior in most cases. Table 13 shows that CNN1 achieves more than 80 % accuracy while CNN2 which also tuned some convolutional layers are slightly inferior. However these results are highly influenced by the single data split and are not an accurately estimate of the general performance on the BC data set. The distribution of how the CNN's predicts are shown in table 15 in Appendix where it is clear that the models trained on the balanced data set are more likely to predict a signal to be "not_usable" than the models trained on the imbalanced training set. Additionally CNN1 mainly seems to learn to predict well on the "usable" class, whereas CNN2 achieve much higher quantity on true negatives. This could indicate that tuning some of the convolutional layers might result in a better feature extraction of artifacts in EEG-spectrograms. Based on this CNN2 with learning rate 0.0001 trained on a imbalanced training set had most correct predictions on the "not_usable" category, see table 16 in Appendix 8.5.

Model	Accuracy	CI	Accuracy	CI	Accuracy	CI
CNN1	0.82	[0.81, 0.83]	0.82	[0.81, 0.82]	0.82	[0.81, 0.83]
CNN2	0.75	[0.74, 0.76]	0.80	[0.79, 0.80]	0.80	[0.79, 0.80]
Learning rate	0.0001		0.0005		0.001	

Table 13: CNN accuracies on imbalanced training set

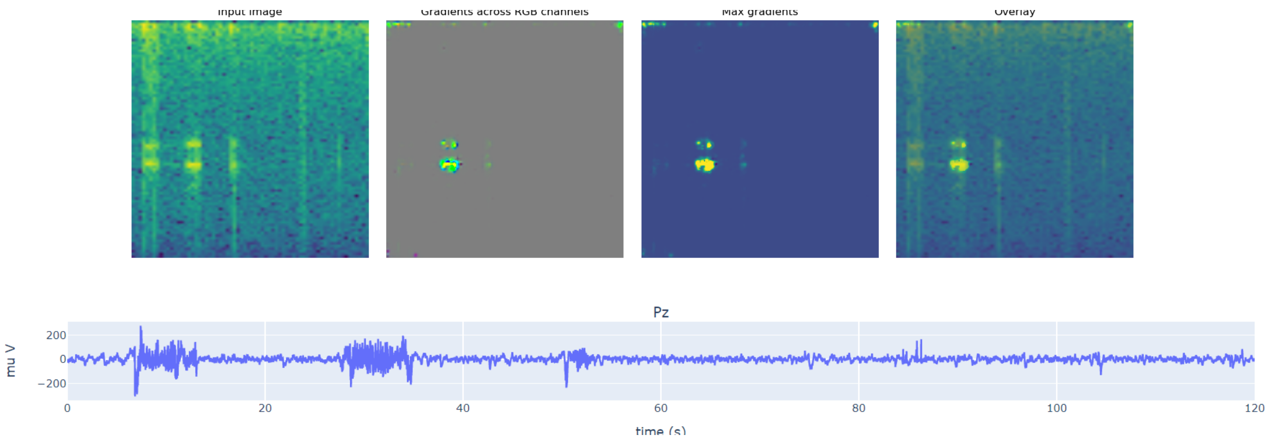
Model	Accuracy	CI	Accuracy	CI	Accuracy	CI
CNN1	0.76	[0.75, 0.77]	0.68	[0.67, 0.69]	0.69	[0.68, 0.70]
CNN2	0.69	[0.68, 0.70]	0.68	[0.67, 0.69]	0.66	[0.65, 0.67]
Learning rate	0.0001		0.0005		0.001	

Table 14: CNN accuracies on balanced training set

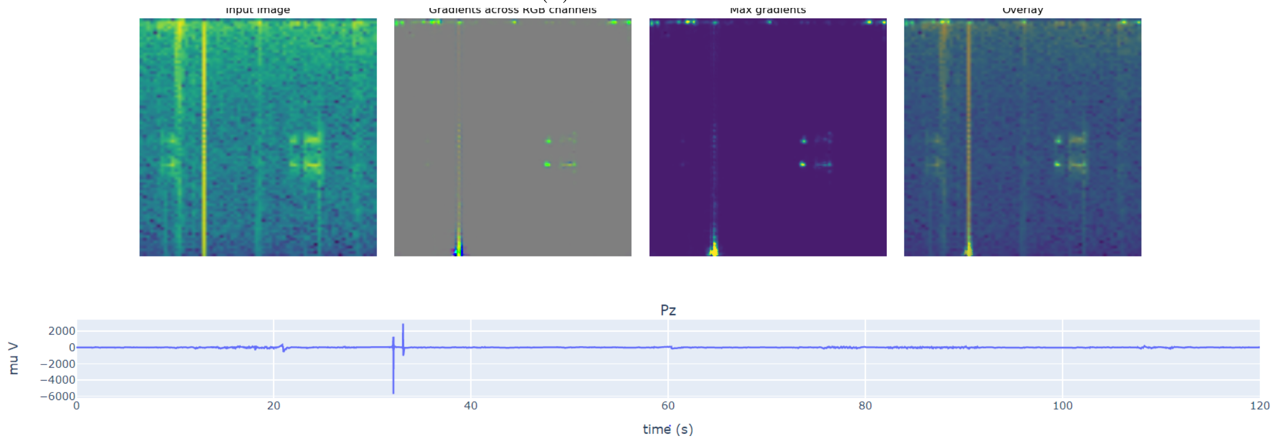
5.6 Saliency mapping

The former mentioned CNN2 in section 5.5 was used to generate new feature vectors, referred to as the abnormal data set which only consists of correctly predicted 'not_usable' spectrograms as described in section 4.7. These spectrograms was investigated with saliency mapping for the chosen CNN. In figure 5.6.24 and 5.6.25 the saliency maps of selected spectrograms along with the raw EEG signal are shown. Other examples of saliency maps are shown in figure 8.5.9 found in Appendix 8.5. (For an unknown reason the toolkit used for generating the plots flipped the spectrograms). The CNN seems to focus on the odd fragments within the spectrograms, primarily vertical trends in form of major spikes as in figure 5.6.24b and 5.6.25a. The gradients for major spikes are primarily visible in the bottom of the spectrograms, indicating that only spikes with full range (top to bottom) is not emphasized other than in the top of the spectrogram (example in figure 8.5.9b in Appendix).

It is noticeable that the top corners (bottom corners originally, at low frequencies) draw attention to the CNN in all saliency maps. In some way it checks if the spectrogram contains high energy at the beginning and the end of the window. Additionally horizontal patterns are found, which originate from the fluctuations in the raw EEG signal, see figure 5.6.24a. In addition it can be noticed that the CNN rarely focus on what is going on in the middle of the spectrogram, with only horizontal blocks being the exception. These however are also very eye catching possible artifacts in the signals, which could stem from muscle movements.



(a) Observation 1

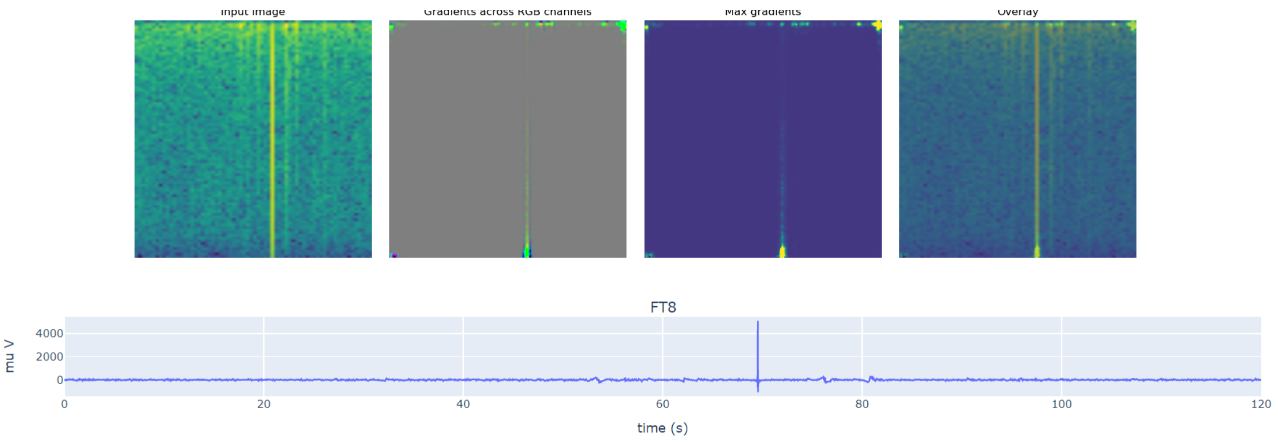


(b) Observation 2

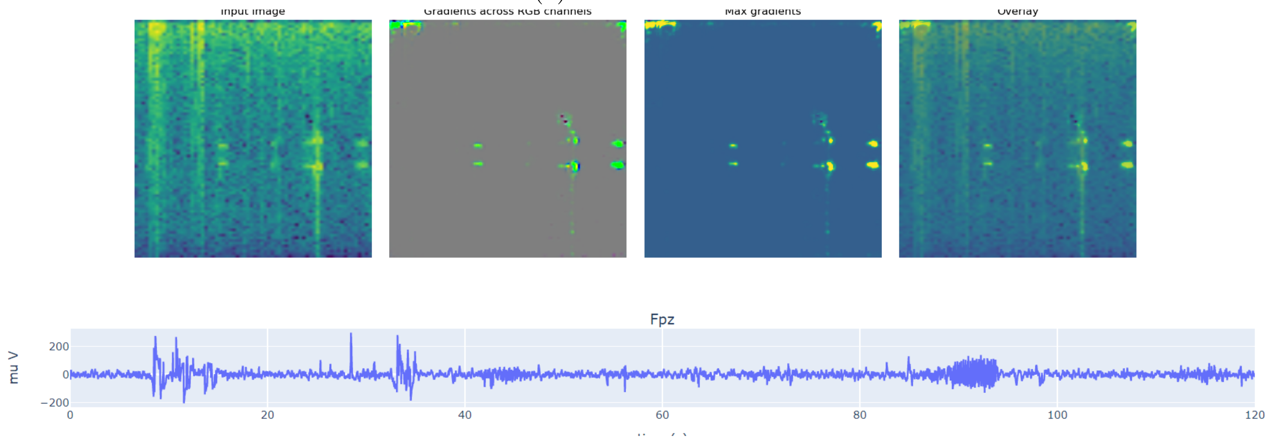
Figure 5.6.24: Saliency maps of spectrograms with corresponding raw EEG-signals. The x-axis on the spectrograms is the time steps and the y-axis the frequency in Hz

The CNN seems to be have some limitations in the number of pixel areas in the spectrogram that have high gradients. Figure 5.6.24a and 5.6.25b the CNN 'misses' the fluctuations in the beginning of the signal, and only weights the major fluctuation later in the signal. This could influence the respective latent feature vectors which may not contain information about these fluctuations that the full spectrograms have, supporting the results seen regarding research question 2.

Especially for the major spikes or horizontal blocks these spectrograms seems to contain evidence of being "not_usable" for medical purposes, although expert knowledge is needed to draw any kind of conclusion on this. For these spectrograms the CNN mainly weights the same pixels areas as humans would notice, which indicate that relevant features can be extracted from the tuned VGG16 which supports hypothesis 3.



(a) Observation 3



(b) Observation 4

Figure 5.6.25: Saliency maps of spectrograms with corresponding raw EEG-signals. The x-axis on the spectrograms is the time steps and the y-axis the frequency in Hz

The main implication of the saliency maps are:

- Only major spikes in the spectrograms are emphasized by the CNN.
- Horizontal blocks draws attention to the CNN corresponding to big fluctuations in the raw EEG signal.
- The CNN seems to focus on the bottom and especially the bottom corners of the spectrograms.
- Less dominating features in the spectrograms are ignored.
- The CNN seems to emphasize some of the same clear features as a human would from looking at the spectrograms.

5.7 Clustering in the latent space projected onto PCA

The GMM fitted with four centroids to the PCA projection of the abnormal data set is shown in figure 5.7.26.

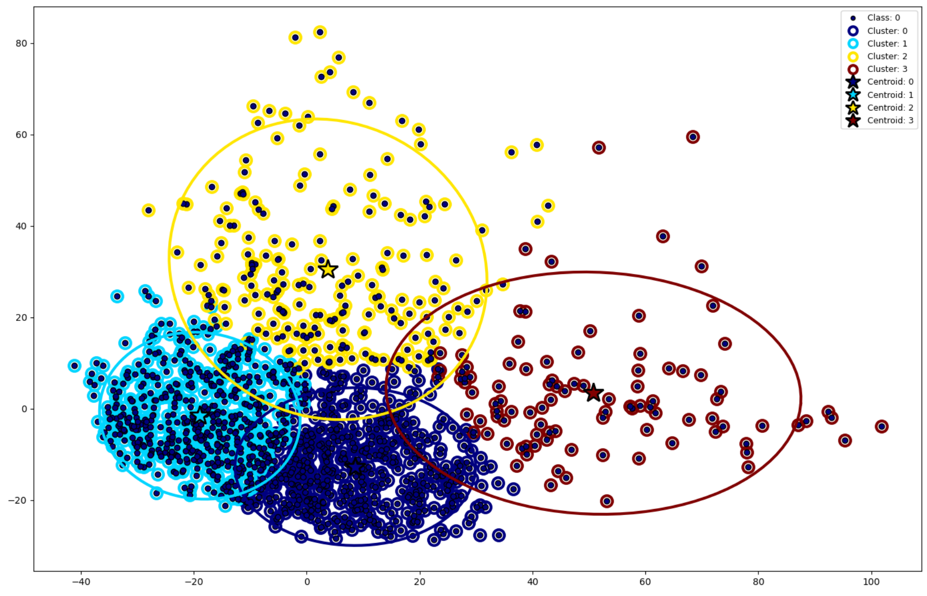


Figure 5.7.26: Gaussian Mixture Model on feature vectors projected onto principal components

The PCA projection in figure 5.7.26 shows that windows (single channels) are to some degree separated in clusters in 2D with a majority in the two blue clusters. In figure 5.7.27 the same PCA projection with labels being the different recordings shows that the different recordings are mixed indicating that recordings belonging to individual does not create groupings.

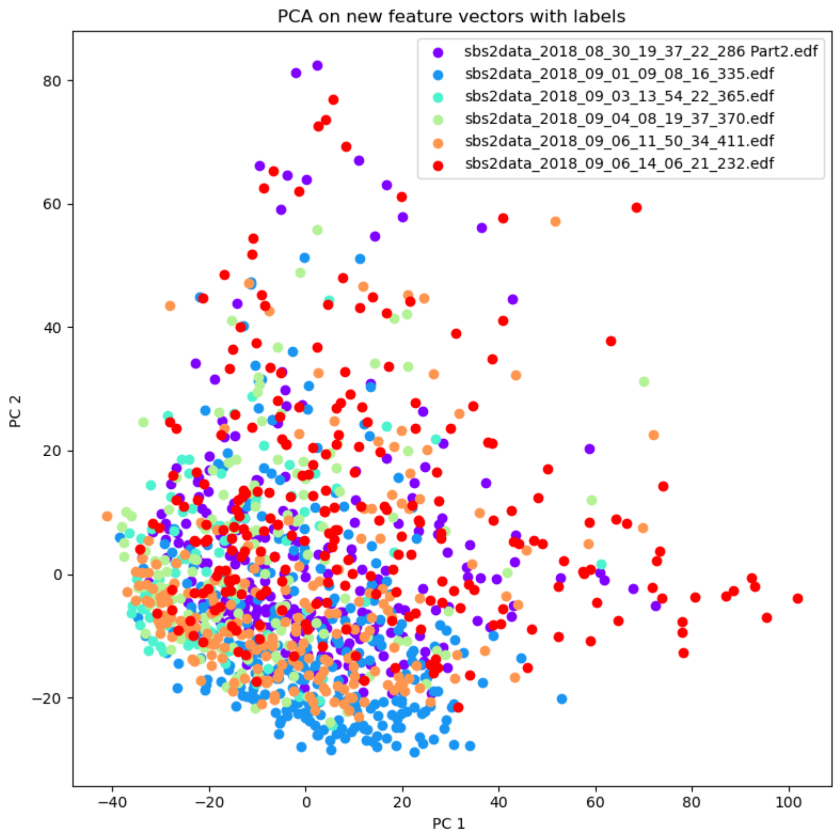
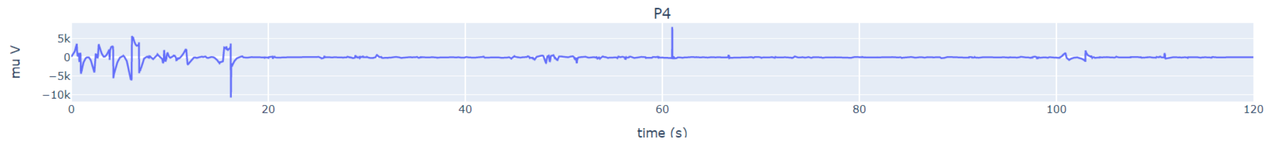
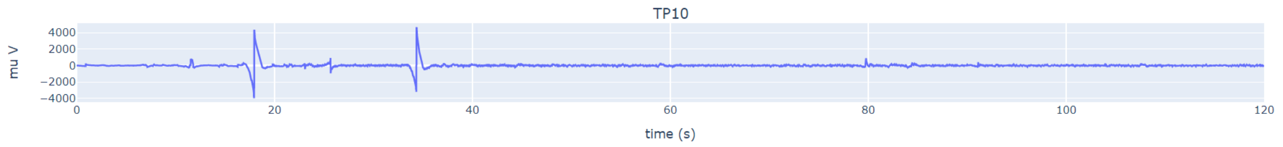


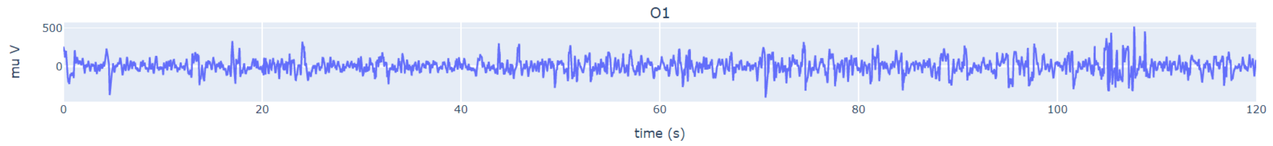
Figure 5.7.27: PCA on new feature vectors with labels being the different recordings

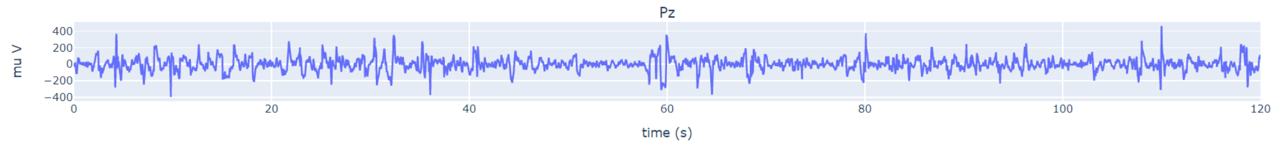
The raw EEG-signals corresponding to four different observations, two from the yellow and two from the dark blue clusters from the GMM in figure 5.7.26, are visualized in figure 5.7.28. Four raw EEG-signals for each of the clusters can be found in appendix figure 8.6.10, 8.6.11, 8.6.12 and 8.6.13. Even though the EEG-signals in the yellow cluster seems to be rather flat, the fluctuations are massive as seen on the y-axis range. The same kind of big fluctuations appear in the red cluster (figure 8.6.10) but with more noise in general. For the two blue clusters, the dark blue seems to be quite noisy throughout a window with a much lower range as in figure 5.7.28, where the turquoise seems to have some underlying noise at lower range of voltage but with few steep fluctuations. The yellow and dark blue cluster are furthest apart in principal component two and could indicate that this principal component reflects the presence of major spikes. In order to validate / invalidate hypothesis 4 regarding being able to identify different groupings of artifacts expertise in EEG signals are needed.





(a) Two observations from the yellow cluster





(b) Two observations from the dark blue cluster

Figure 5.7.28: Two raw EEG signals from each of the yellow and dark blue cluster (cluster 2 and 0 in figure 5.7.26)

6 Discussion

6.1 Group structures in data

In section 5.1 it was visible from the PCA and t-SNE plots of the spectrograms and the feature vectors that windows from the same recording were clustered together. The cross validation also showed that the accuracy within the folds seemed to vary a lot, indicating some dependency between windows from the same recording. The statistical methods used to evaluate the results assumes independence between the windows, and will systematically yield too narrow confidence intervals if windows from the same recordings are in fact not independent. This will lead to wrong conclusions about which models and representations that perform significantly different.

Although there are many implications that ML-models trained on spectrograms can beat a baseline, supporting hypothesis 1, further research is needed in order to generate confidence intervals that incorporates the dependency between windows. A way to handle the dependency could be to classify recordings instead of windows, which will be discussed later in section 6.7.

6.2 Confounders

The group structures also make it necessary to think about confounders related to the persons beings recorded, that influence both the label of the recording and the EEG-signal. This could for example be the age of the person being recorded. In section 2.1 it is mentioned that the EEG-signal is dependent on age, however it also seems reasonable to think that whether your able to uphold a resting state to get a recording that is fit for clinical trials, can be very age dependent. This could potentially create a bias in the ML-models, in which EEG from children is systematically being predicted to be not-usable.

Another important aspect is whether recordings of patients with epilepsy have more unwanted artefacts. If this is the case the same bias as described above might occur, resulting in rejection of EEG with elliptical discharges. However the labeling of the Brain-Capture recording also includes labels of whether the EEG shows signs of elliptical discharges, so this could potentially be taken into account, to avoid this bias.

6.3 Problems with labeling due to class-imbalance

When labeling the EEG-windows all windows receives a label according to the recording that they originate from as described in section 4.3. However in the visualization it was observed that recordings labeled "usable" might contain windows with not-usable EEG and vice versa. This leads to windows that are labeled incorrectly. In the data set 80 % of the recordings are labeled "usable" and 20 % are labeled "not_usable". In the scenario in which a machine learning model can perfectly discriminate between windows that are truly not usable and truly usable, the recordings labeled "not_usable" has to contain more than 4 times as many truly not usable EEG-windows, as the recordings labeled "usable". If this is not true, more than half of the truly not-usable EEG-windows are contained in the recordings labeled "usable" and will be labeled wrong, which makes it impossible for a machine learning model to perform better than baseline. Even though the extremeness of this example makes it unrealistic, it outlines some of the problems with the labeling strategy used in this report, and some milder version of the described example seems plausible. It could explain why the classifiers on both representations have an accuracy close to baseline.

6.4 Data representation

There are several plausible reasons for why the classification performance seems better on the spectrogram representation compared to the feature representation.
First off, one can discuss the advantages and disadvantages of transfer learning, and what happens when a pre-trained VGG16 model generates its latent representation of the spectrograms. In image recognition transfer learning can be used because low level features are typically shared among different tasks. These shared features is typically considered to be edges, geometric patterns etc. If none or just a small subset of such features are shared between tasks, it may require additional tuning of the transferred weights to learn meaningful representations. This may be the case in our project due to the distinct appearance of spectrograms. A spectrogram does not share a lot of features with the classes in the ImageNet data set, rather they contain a quite specific range of primarily vertical and horizontal patterns. If not trained to keep these features the CNN might might loose important information when creating feature maps or in the pooling process.
It is possible to extract a latent representation at each of the VVG16's 16 layer and for each layer different information and abstract concepts may be present or filtered away. It is an open question which latent representation from the VGG16 that is the best. Hence it might have been interesting to test different latent representations from different. But as mentioned in the earlier in the method section it was decided to use as much of the VGG16 network possible, in hope of achieving a meaningful and dimensionally reduced representation. It was decided that it was outside the scope of this project to investigate classification performance on several different latent representations. An important advantage of using most of the VGG16 to produce feature vectors is the resulting dimensions. If the latent representation was extracted earlier a very significant increase in dimentionality of the feature representation and therefore also in computational complexity in the subsequent processing.

As mentioned earlier, the feature representation is generated from a pretrained VGG16 without any further tuning. As visualized in figure 6.4.29 it appears that the original VGG16 only focuses on very small part of the spectrogram. In comparison the tuned version of the VGG16 (which had several layers trained), seem to notice several areas, that we as humans also would regard as important. It is yet to be seen if feature vectors generated from a tuned VGG16 network would be superior to the spectrogram representation.

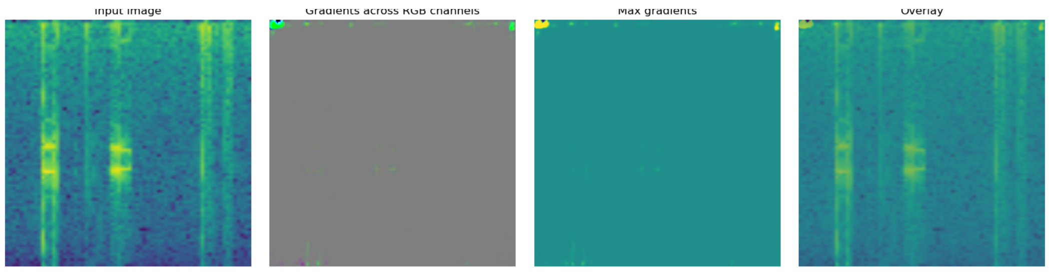


Figure 6.4.29: Saliency map of a spectrogram generated with VGG16 without any additional tuning of weights. From left to right is the raw spectrogram (upside down), gradients across RGB, Max gradients, and an overlay.

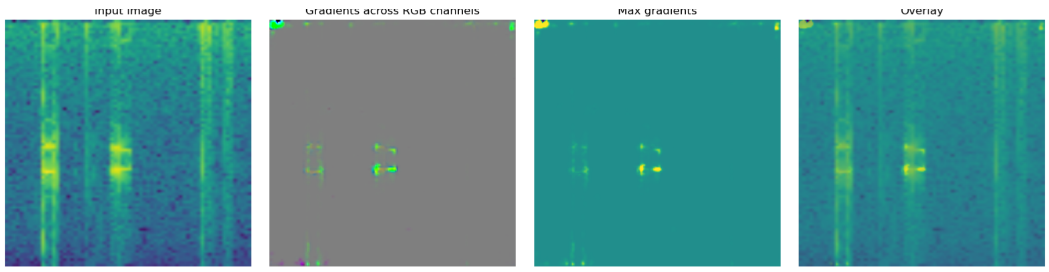


Figure 6.4.30: Saliency map of a spectrogram generated with VGG16 without any additional tuning of weights. From left to right is the raw spectrogram (upside down), gradients across RGB, Max gradients, and an overlay.

6.4.1 An alternative interpretation

The rejection of hypothesis two in the results was based on a classification task in which the labeling of the observations was questionable. In the visualization it was seen that clusters with windows labeled not usable, did not necessarily contain any strong artefacts 4.3. It was also seen that outliers in the PCA-plot for the feature vectors actually did seem to contain strong artefacts, which begs the question of whether it is the labeling rather than the feature vector representation that is not meaningful. In the visualization it was also seen that the spectrogram representation was heavily influenced by the group structures due to windows originating from the same recording. It could be that the spectrogram representation can outperform a baseline because it classifies based on confounders as described in section 6.2. Artefacts like eye-blinks, muscle contraptions or line-noise, are mostly independent of the group structures related to the recordings, so it could be that unsupervised outlier methods would be superior on feature vector representations rather than spectrograms at detecting non-usable EEG.

6.5 Handling class imbalance

The starting point for the classification problem was a imbalanced training set. To handle this more windows of not-usable EEG was included to create a balanced training set which resulted in lower accuracy's (for classifiers on feature vectors) which was expected. However when looking at the confusion matrices of the models trained on the balanced the models still predict more windows labeled usable correctly proportionally than the not-usable windows. However the AUC scores are the same, indicating that the separation between the classes is still the same when training on a balanced dataset. This indicates that the class not-usable is more difficult to predict than usable, and the models are therefore more biases towards the usable class. Intuitively, it makes sense that usable EEG-windows are more alike than not-usable windows, which could be the reason for this bias. Since we are more interested in predicting not-usable windows correct, more should be done in order to account for this bias. One way could be to include even more windows of the not-usable creating a training set that is bias towards the not-usable class. Another way could be to make data augmentation on the not-usable class, creating more not-usable observations to include.

6.6 Unsupervised clustering in the latent space

Without any expert knowledge in recognising characteristics of raw EEG signals, it is difficult to draw any definitive conclusion about whether meaningful clustering of 'not_usable' windows are obtained. However some apparent group tendencies was noticed about the range of amplitude and general fluctuations. But even noticing these tendencies should be treated with caution. If looking at figure 6.6.31 the range of the unscaled raw EEG signal might trick one into believing that nothing really goes on for a long duration. If looking at the scaled raw EEG signal there appears to be some general noise with major fluctuations in the beginning and in the middle. When looking at figure 5.7.27 and figure 5.7.28 we see that the yellow cluster have far larger amplitude fluctuations when compared to the dark blue. Which means that the clustering seems to demonstrate the ability to detect large fluctuations. However it seems to be very general features in the EEG that could potentially be handled manually, without using the complex feature representation of the VGG16.

This illustrates the complex task at hand, since a raw EEG signal contains a lot of information along with what appears to be noise. In order to evaluate the unsupervised clustering, labels on different types of artifacts would be needed. This could serve as an alternative way to investigating research question 2 regarding whether a latent representation is suited for spectrograms of EEG signals.

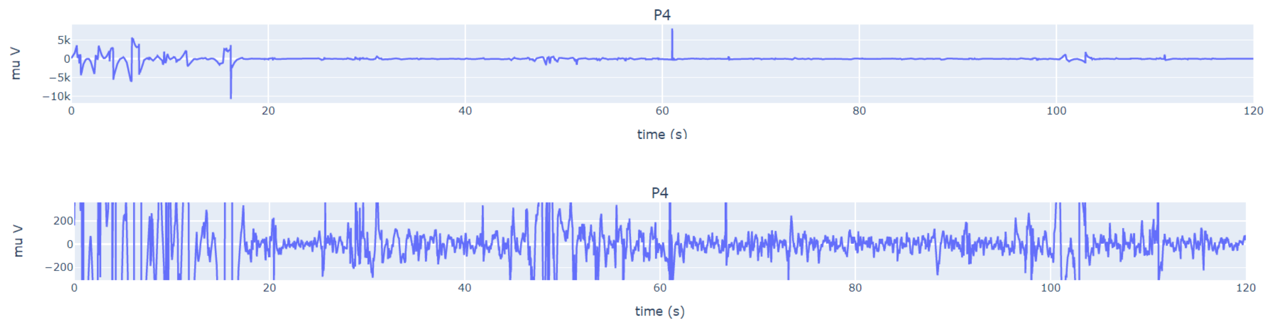


Figure 6.6.31: Some noisy outliers before and after adjusting the y-axis

6.7 Classifying recordings instead of windows

An alternative assumption to the one stated in section 4.3, regarding the labeling of the windows might be that a recording labeled "not-usable" have more windows with "not-usable" EEG than a recording labeled usable. This would make it possible to make predictions about whether a full EEG recording is usable for clinical use. Rejecting recordings before they are sent for visual inspection by medical personnel, would most likely also be valuable for Brain-Capture. This could be implemented and tested in this study's experimental setup in a simple way, by fitting a threshold to the number of windows predicted to be unusable for a full recording to be unusable, in each training split. This approach could then be validated on the test splits. More sophisticated approaches, that could take the time-stamp of the windows into account could also be applied. An example could be recurrent neural networks, that can handle information sequentially, and is therefore well suited for time-series analysis.

6.8 Preprocessing

In preprocessing many choices were made after what is considered best practise in EGG signal processing, either by our academic advisor or by [4]. Other filters could have been considered and may have given a different result, especially since our task is to identify outliers and they may disappear in the filters. On the other hand if an outlier can be removed by a standard filter is it then worth detecting? The choice of using 120 seconds EEG per window is also just a compromise between information in each picture not getting to compact and how much data we want. Some artefacts are only present in small time-intervals at a time, in which the pattern of this artefact might be lost in a 120-seconds long spectrogram with 1.75 seconds time-steps. Alternatively a dimensionality reduction of the original raw EEG-signal could also have been tested as an representation, or pictures of raw EEG passed through VGG16.

6.9 Limitations regarding data set

The provided data set posed several challenges when used for the purpose in this project. As a result there are a lot of important decision to make before the data is ready for machine learning purposes. Furthermore is it a memory wise large data set, that for this reason can be challenging to process. This section will briefly discuss some of the most important limitations and challenges encountered when handling the data set.

6.9.1 No annotations for individual windows

For each EEG-recording annotations are provided for the whole file. This begs the question of whether or not each window is a good representative of this annotation. The underlying cause for a recording to be labelled as non-usable by an expert neurologist is unknown, and hence it may differ in several ways. There are many plausible reasons for why a recording should be given a certain label, and some of these causes could possibly be present in only part of the recording, and for this reason only a part of windows may be good representative for the class of which it belongs. Other causes such as background noise from electrical devices or ill-placed medical equipment should be visible in all or most windows, and hence it seems feasible that

some causes can be learned by classifiers.
A possible way to handle this may to take all of the windows into account instead of labeling each window individually. An approach that may be feasible is discussed in section 6.7.

6.9.2 Annotation quality

Naturally labeling performed by different neurologists are subject to some personal bias and as mentioned in the datas et section 2.3 there were 11 different expert neurologists that evaluated the 126 files that have been used. Each neurologist has evaluated between 1-26 files, and it is unknown if these neurologists are familiar with the format of EEG-recordings with only 14 channels. This may results in extra variability in the data set and disagreeing experts may directly effect the achievable performance for classifiers.
Labeling by several neurologists for each file would have been preferable as this would increase the validity of the individual labels. As in many other cases machine learning tasks, the quality and consistency of annotations sets a natural bound of the accuracies classifiers can achieve, and with a signal as complex as EEG, some disagreement is unavoidable.

6.9.3 Lack of labels

A critic this project could receive is that only a portion of the data given is used. Usually more data is better, and there are many interesting ways of using the insufficient labelled data in ways that could help when classifying the usability of EEG signals, however unsupervised methods beyond GMM is outside of the scope of this project. This project has been a part of a larger EEG-signal research team's effort to investigate how machine learning can aid in EEG-diagnosis and other groups have investigated the potential of such unsupervised techniques.

6.10 Computational limitations

6.10.1 Training the classifiers

Limited time and computational resources was a major factor in this project. For the machine learning classifiers trained on spectrograms and feature vectors only the default hyperparameters were used. Even though it would be very extensive to carry out an experiment that evaluates several hyperparameters for several models, new superior models might occur. In order for the results to be more independent of the specific classifier multiple classifiers were included and tested to give implications of the general performance on the data representations. A way computational limitation could be handled is by making a dimensionality reduction on the inputs (spectrograms and feature vectors) using for instance PCA. This would allow for a larger search in the hyperparameter-space, with the trade-off of using a diminished data representation

Computational- or time related restrictions also applies for the training of the CNN. Some choices of hyper-parameters were made initially, like the Adam optimizer, the number of epochs (2), batch size (256) and the limited amount of windows (20) per file.

6.10.2 Training a CNN

There were a number of large data sets on EEG-signals available to us fx. The Temple University Hospital data set (TUH) which has +10.000 different recordings. However within the boundary of the computational resources and time frame, training a CNN from scratch was not feasible. Additionally it was not certain that a new CNN would be able to generalize from the rather clean TuH data (64 channels performed under clinical conditions) to the noisy BC data set. Besides taking advantages of the TuH data set available, the input could be altered to handle all 14 channels and multiple windows. For the same reasons the performance of the CNN is not comparable to the machine learning algorithms since, the CNN makes predictions based on one channel. However to use transfer learning of a pretrained VGG16 the input dimensions (single channel EEG-spectrograms) must be kept in order to exploit its feature extraction.

Due to time-frame only a small set of new feature vectors were generated for clustering. It remains to be seen if the latent space of CNN2 with a learning rate of 0.0001 that used feature learning could significant improve the performance of classifiers.

6.11 Ethical considerations

All the data in this project has been anonymized by the creators of the data set in order not to compromise any personal information. The scope of the Brain-Capture project is to close the treatment gap between developed and under-developed countries. In that way the scope of this project seems to serve a good ethical purpose, using AI as a mean in this process.

7 Conclusion

This project revolved around classification of EEG-windows' clinical usability. By creating two different data representation and comparing several machine learning models performance on each of the representations, we were able to find evidence that indicated support for our first hypothesis, which was that it is possible to outperform a baseline model in classifying EEG-windows for their clinical usability. However a correction of the dependency between windows due to the group structures in the data, has to be made for further confirmation. The evidence presented in the results section did not suggest that the latent representation created with the use of a pre-trained VGG16 allowed for models to obtain superior classification. However several concerns were raised in the discussion about the label assumptions and quality, which may hinder any definitive conclusions on the matter. Through saliency it was possible to get an indication of which features in the EEG that the CNN emphasized when making predictions about the clinical usability of a EEG window. However only very dominant features were emphasized and not all seemed relevant from a human perspective. Through clustering it seemed that some artefacts from the 'not-usable' EEG shared characteristics, however further evaluation from experts is needed to evaluate the meaningfulness of these characteristics.

References

1. Kjems, U., Hansen, L. K. & Strother, S. C. Generalizable singular value decomposition for ill-posed datasets. *Advances in Neural Information Processing Systems.* ISSN: 10495258 (2001).
2. Maaten, L. v. d. & Hinton, G. Visualizing Data using t-SNE. *Journal of Machine Learning Research* **9,** 2579–2605. ISSN: ISSN 1533-7928 (2008).
3. Wang, B., Wan, F., Mak, P. U., Mak, P. I. & Vai, M. I. Outlier detection for single-trial EEG signal analysis. *2011 5th International IEEE/EMBS Conference on Neural Engineering, NER 2011,* 478–481 (2011).
4. Sanei, S. & Chambers, J. A. *EEG Signal Processing* ISBN: 9780470025819 (John Wiley and Sons, May 2013).
5. Simonyan, K. Deep Inside Convolutional Networks : Visualising Image Classification Models and Saliency Maps arXiv : 1312 . 6034v2 [cs . CV] 19 Apr 2014, 1–8 (2013).
6. Bigdely-Shamlo, N., Mullen, T., Kothe, C., Su, K.-M. & Robbins, K. A. The PREP pipeline: standardized preprocessing for large-scale EEG analysis. *Frontiers in Neuroinformatics* **9,** 1–19. ISSN: 1662-5196. `http://journal.frontiersin.org/Article/10.3389/fninf.2015.00016/abstract` (June 2015).
7. Frølich, L., Andersen, T. S. & Mørup, M. Classification of independent components of EEG into multiple artifact classes. *Psychophysiology* **52,** 32–45. ISSN: 14698986 (2015).
8. Korff, C. M. Wyllie's treatment of epilepsy: principles and practice. *Epilepsy & Behavior* **52,** 8. ISSN: 15255050. `https://linkinghub.elsevier.com/retrieve/pii/S1525505015004928` (Nov. 2015).
9. Simonyan, K. & Zisserman, A. *very deep convolutional networks for large-scale image recognition* in (2015). `http://www.robots.ox.ac.uk/`.
10. Takahashi, S. *et al.* Focal frontal epileptiform discharges in a patient with eyelid myoclonia and absence seizures. *Epilepsy & Behavior Case Reports* **4,** 35–37. ISSN: 22133232. `https://linkinghub.elsevier.com/retrieve/pii/S2213323215000250` (2015).
11. Ian Goodfellow, Yoshua Bengio & Aaron Courville. *Deep Learning* 2016. `http://www.deeplearningbook.org/contents/convnets.html`.
12. Shin, H. C. *et al.* Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning. *IEEE Transactions on Medical Imaging* **35,** 1285–1298. ISSN: 1558254X (May 2016).
13. Tandle, A., Jog, N., D'cunha, P. & Chheta, M. Classification of Artefacts in EEG Signal Recordings and EOG Artefact Removal using EOG Subtraction. *Communications on Applied Electronics* **4,** 12–19. ISSN: 23944714. `http://www.caeaccess.org/research/volume4/number1/tandle-2016-cae-651997.pdf` (Jan. 2016).
14. Mckenzie, E. D. *et al.* Validation of a smartphone-based EEG among people with epilepsy: A prospective study. *Scientific Reports* **7,** 1–8. ISSN: 20452322 (Apr. 2017).
15. Liu, J. *et al.* Anomaly detection for time series using temporal convolutional networks and Gaussian mixture model. *Journal of Physics: Conference Series* **1187.** ISSN: 17426596 (2019).
16. Mørup, M., Schmidt, M. N. & Herlau, T. *Introduction to Machine Learning and Data Mining* ISBN: 9781420011784 (2019).

17. Williams, J. A. *et al.* Smartphone EEG and remote online interpretation for children with epilepsy in the Republic of Guinea: Quality, characteristics, and practice implications. *Seizure* **71,** 93–99. ISSN: 15322688 (Oct. 2019).
18. Chen, G., Lu, G., Xie, Z. & Shang, W. Anomaly detection in EEG signals: A case study on similarity measure. *Computational Intelligence and Neuroscience* **2020.** ISSN: 16875273 (2020).
19. Sokolov, E. *et al.* Tablet-based electroencephalography diagnostics for patients with epilepsy in the West African Republic of Guinea. *European Journal of Neurology,* ene.14291. ISSN: 1351-5101. https://onlinelibrary.wiley.com/doi/abs/10.1111/ene.14291 (May 2020).
20. Antoniades, A., Spyrou, L., Took, C. C. & Sanei, S. *Deep learning for epileptic intracranial EEG data* tech. rep. ().
21. Bengio, Y., Courville, A. & Vincent, P. *Representation Learning: A Review and New Perspectives* tech. rep. (). http://www.image-net.org/challenges/LSVRC/2012/results.html.
22. *Convolution - Wikipedia* https://en.wikipedia.org/wiki/Convolution?fbclid=IwAR17LuZ38Ba025u0w84lHeEaQ7snnCz46UQ6BKfgkdcVdUuPuBuoFRnc59c.
23. Emilie, A., Wedenborg, J., Mortensen, A. E., Hejgaard, B. K. & Kjaersgaard, I. F. *Novelty detection in electroencephalography (EEG) using a two-layer convolutional neural network* tech. rep. (). http://www.michaelshell.org/contact.html.
24. *Epilepsy* https://www.who.int/health-topics/epilepsy#tab=tab_1.
25. *examples/main.py at 97304e232807082c2e7b54c597615dc0ad8f6173 · pytorch/examples* https://github.com/pytorch/examples/blob/97304e232807082c2e7b54c597615dc0ad8f6173/imagenet/main.py#L197-L198.
26. *Mayo Clinic Epelepsy* https://www.mayoclinic.org/diseases-conditions/epilepsy/diagnosis-treatment/drc-20350098?fbclid=IwAR0kqh6kEW3j-qWzvWcgWu77S2UcJu0UtxwSOHE9t
27. *Neural networks and deep learning* http://neuralnetworksanddeeplearning.com/chap6.html.
28. *Uncovering what neural nets "see" with FlashTorch - Towards Data Science* https://towardsdatascience.com/feature-visualisation-in-pytorch-saliency-maps-a3f99d08f78a.

8 Appendix

8.1 Reproducibility

To ensure reproducibility of the results presented in the project, several tools and action has been taken. This includes having a version control with a github repository, setting random seeds, and saving the versions of our scripts that produce the results. The project source code can be downloaded at https://github.com/Madscba/Fagprojekt2020. In the github repository is a requirements.txt file that can be used to construct a virtual environment (venv) which has all the packages needed to run all the scripts of the project and generate the results. A list of these packages is found right after this paragraph. All classifiers are imported from Sci-kit-learn, and implemented with default values. In this project a few special packages in python have been utilized, and here is a brief overview of the most important ones. "mne", is a packages with a lot of tools for exploring, visualizing and analyzing human neurophysiological data, and it has played a big role when preprocessing our raw EEG-data. A more elaborate description can be found at `https://mne.tools/stable/python_reference.html`. "flashtorch" is the toolkit used for saliency mapping, and it works great with pytorch, which is the machine learning framework used to train CNNs, more information on "flashtorch" and "pytorch" can be found at `https://pypi.org/project/flashtorch/0.1.0/#api-guide` and `https://pytorch.org/`. The inteactive plots was made with Plotly for more information see https://plotly.com/python/. The data used for the project cannot be shared without permission, and to gain access one would have to contact us.

Python, ver: 3.8.1, packages used:	
cycler==0.10.0	python-dateutil==2.8.1
decorator==4.4.2	pytz==2020.1
future==0.18.2	PyWavelets==1.1.1
imageio==2.8.0	scikit-image==0.17.2
joblib==0.15.1	scikit-learn==0.23.1
kiwisolver==1.2.0	scipy==1.4.1
matplotlib==3.2.1	setuptools==41.2.0
mne==0.20.5	six==1.15.0
networkx==2.4	sklearn==0.0
numpy==1.18.5	threadpoolctl==2.1.0
pandas==1.0.4	tifffile==2020.6.3
Pillow==7.1.2	torch==1.5.0
pip==19.2.3	torchvision==0.6.0
pyparsing==2.4.7	flashtorch==0.1.3
plotly==4.7.1	seaborn==0.10.0
xlrd==1.2.0	

8.2 Content of the EEG-evaluation for each recording in the BC dataset

- Status - Completed/In progress /Pending
- All leads working throughout recording - Binary
- 'Is Eeg Usable For Clinical Purposes' - Binary
- Epileptiform Discharges Present - Binary
- Normal Eeg Pattern - Binary
- Quality Of Eeg - Numerical
- Recommend Further 10 20 Eeg - Binary (true/false)
- Stage Nrem2 Sleep Architecture Present - 3 options (yes,no unsure)
- Slowing Focality Paracentral - Comments
- Slowing Focality Occipital - Comments
- Slowing Present - 3 options (yes,no unsure)
- Recommend No Change To Patient Management Or Diagnosis - True/False

- Recommend Neuroimaging - True/False
- Epileptiform Discharges Type - Comments
- Epileptiform Discharges Focality Frontal - Comments.
- Epileptiform Discharges Focality Temporal - Rigth/Left + Comments
- Epileptiform Discharges Focality Parietal - Right/left + comments
- Epileptiform Discharges Focality Paracentral - Right/left + comments
- Epileptiform Discharges Focality Occipital - Right/left + comments
- Slowing Type (Generalized or Focal)
- Slowing Focality Frontal - Comment
- Slowing Focality Temporal - comment
- Slowing Focality Parietal - comment

8.3 Confusion matrices's all classifiers

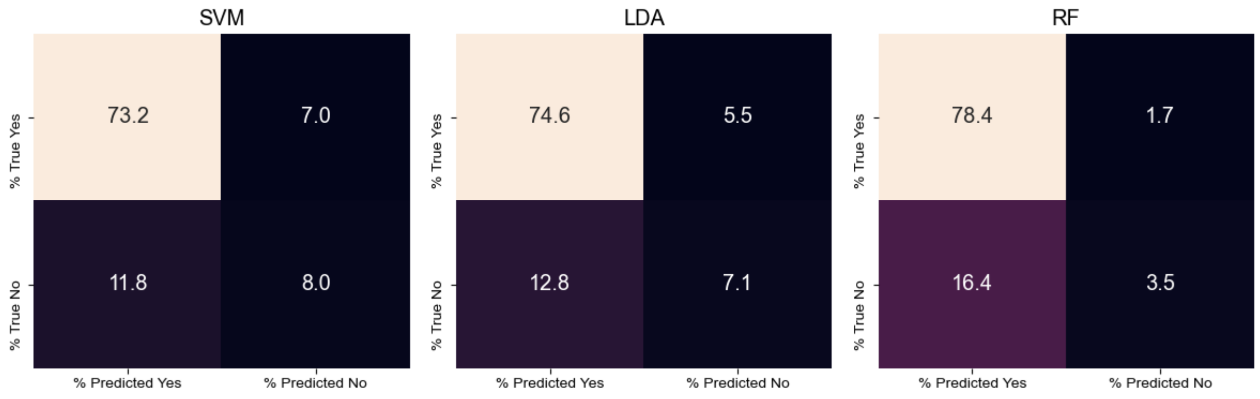


Figure 8.3.1: Confusion matrix for feature vectors unbalanced

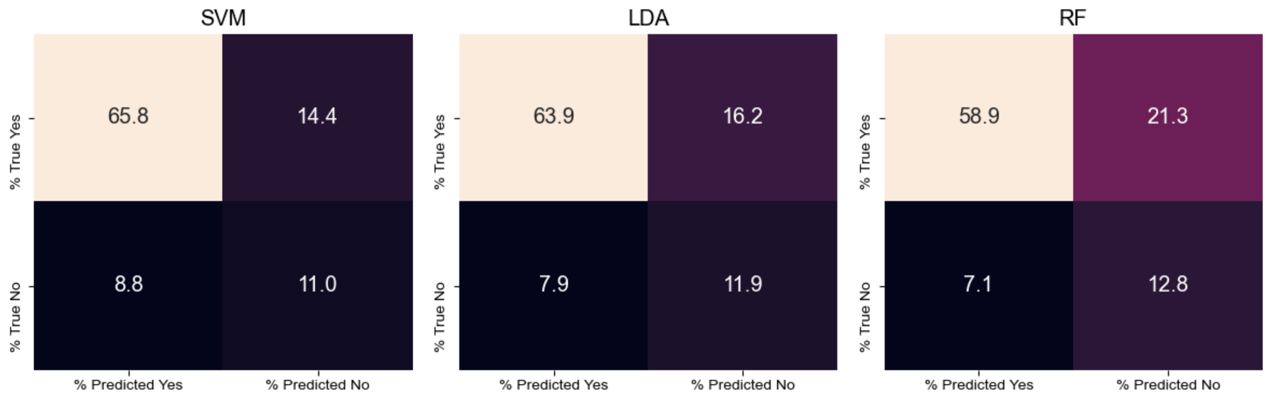


Figure 8.3.2: Confusion matrix for feature vectors balanced

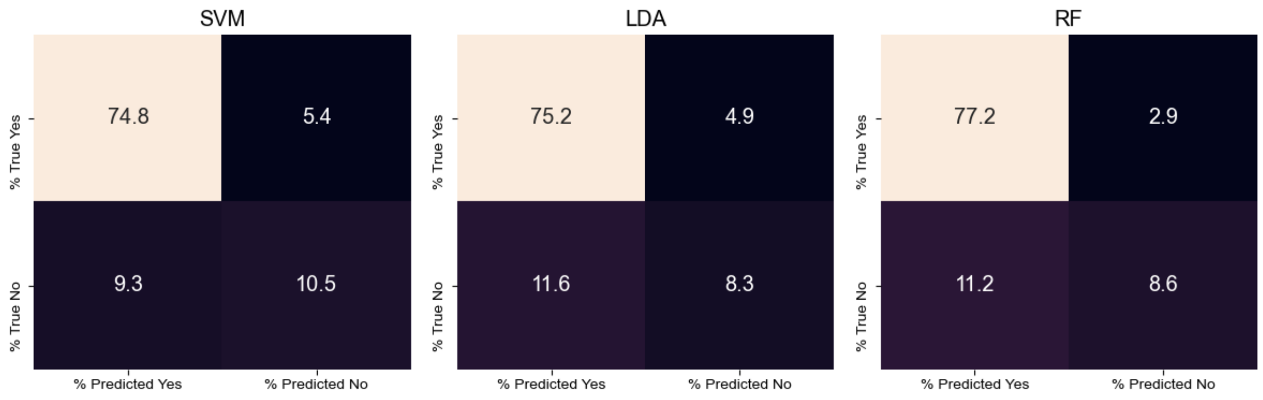


Figure 8.3.3: Confusion matrix for spectrograms unbalanced

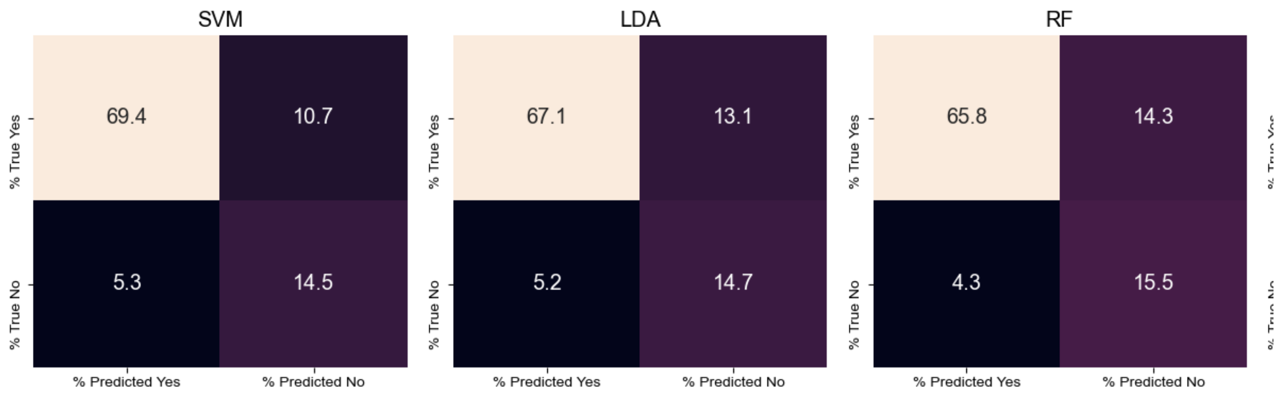
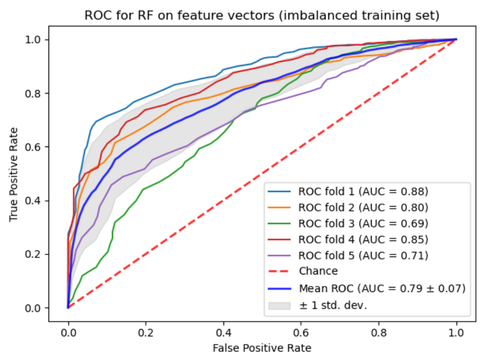
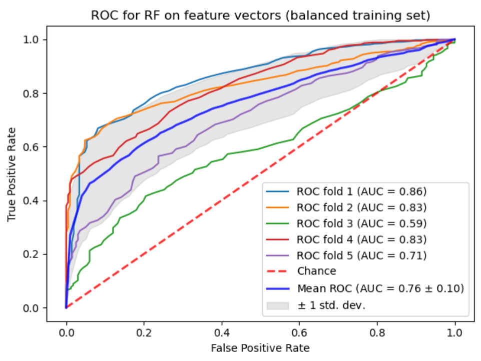


Figure 8.3.4: Confusion matrix for spectrograms balanced

8.4 ROC and AUC

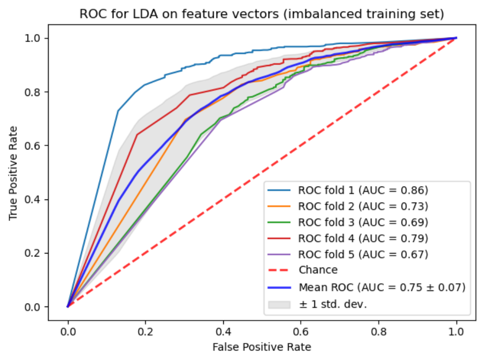


(a)

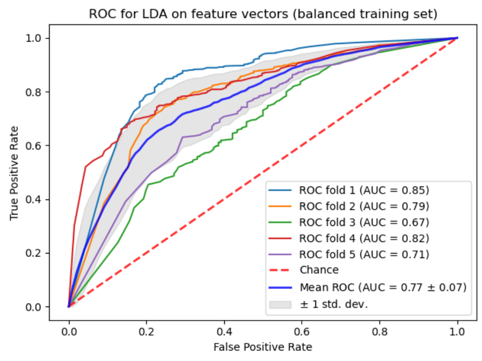


(b)

Figure 8.4.5: ROC for RF on Feature vectors

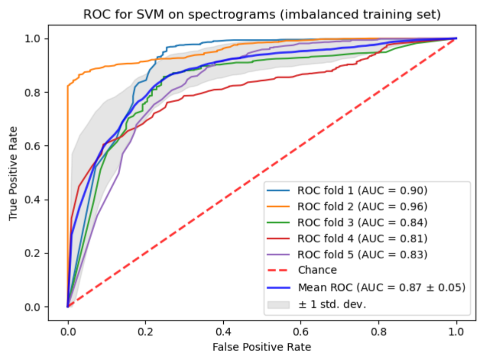


(a)

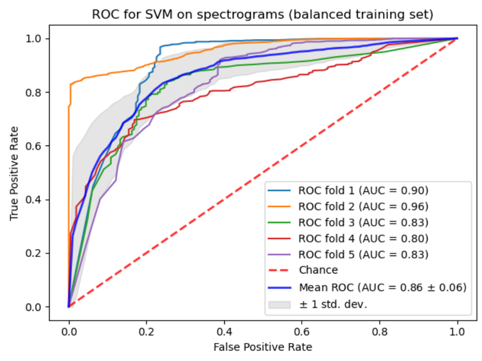


(b)

Figure 8.4.6: ROC for LDA on Feature vectors

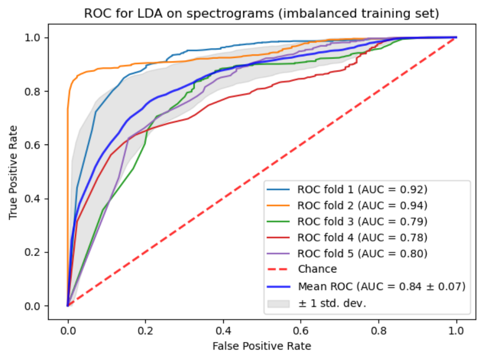


(a)

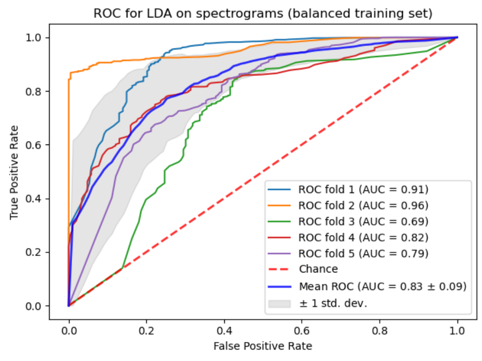


(b)

Figure 8.4.7: ROC for SVM on spectrograms



(a)



(b)

Figure 8.4.8: ROC for LDA on spectrograms

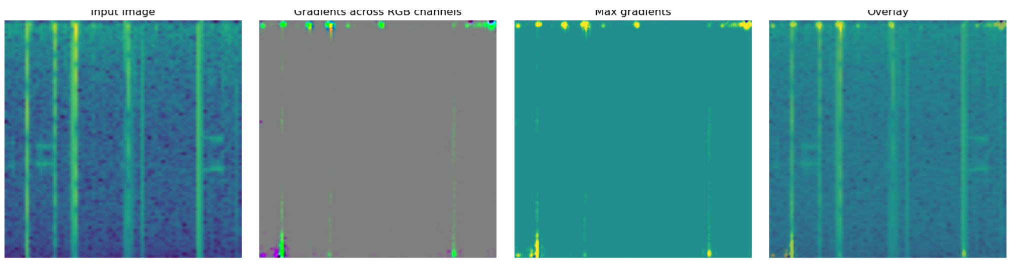
8.5 CNN table

Model	% 'is_usable' predictions	% 'is_usable' predictions
CNN1	0.89	0.64
CNN2	0.83	0.57
Training data	Imbalanced	Balanced

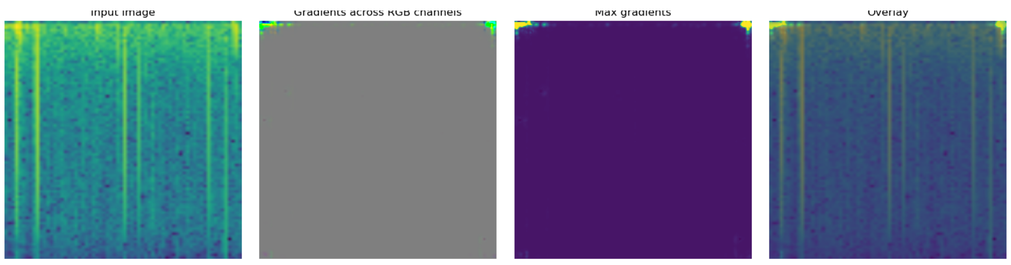
Table 15: Percentage of 'is_usable' predictions across learning rates

Model	TN	TN	TN
CNN1	0.38	0.35	0.31
CNN2	0.60	0.29	0.28
Learning rate	0.0001	0.0005	0.001

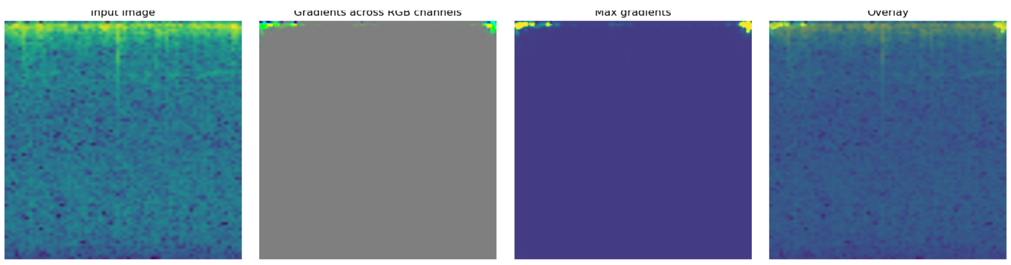
Table 16: % correct predictions in the 'not_usable' class for CNN's trained on imbalanced training set



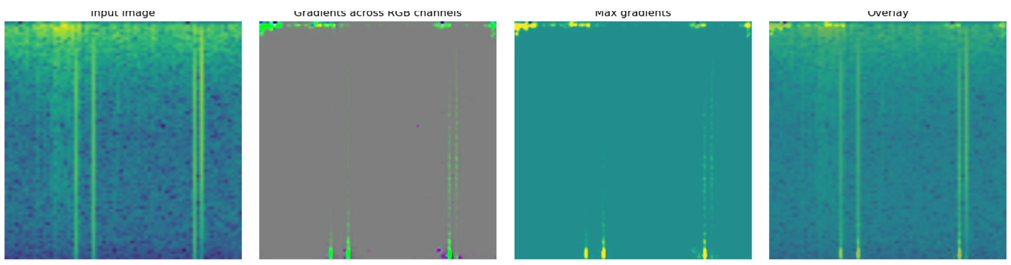
(a)



(b)



(c)



(d)

Figure 8.5.9: Examples of saliency maps

8.6 Clustering

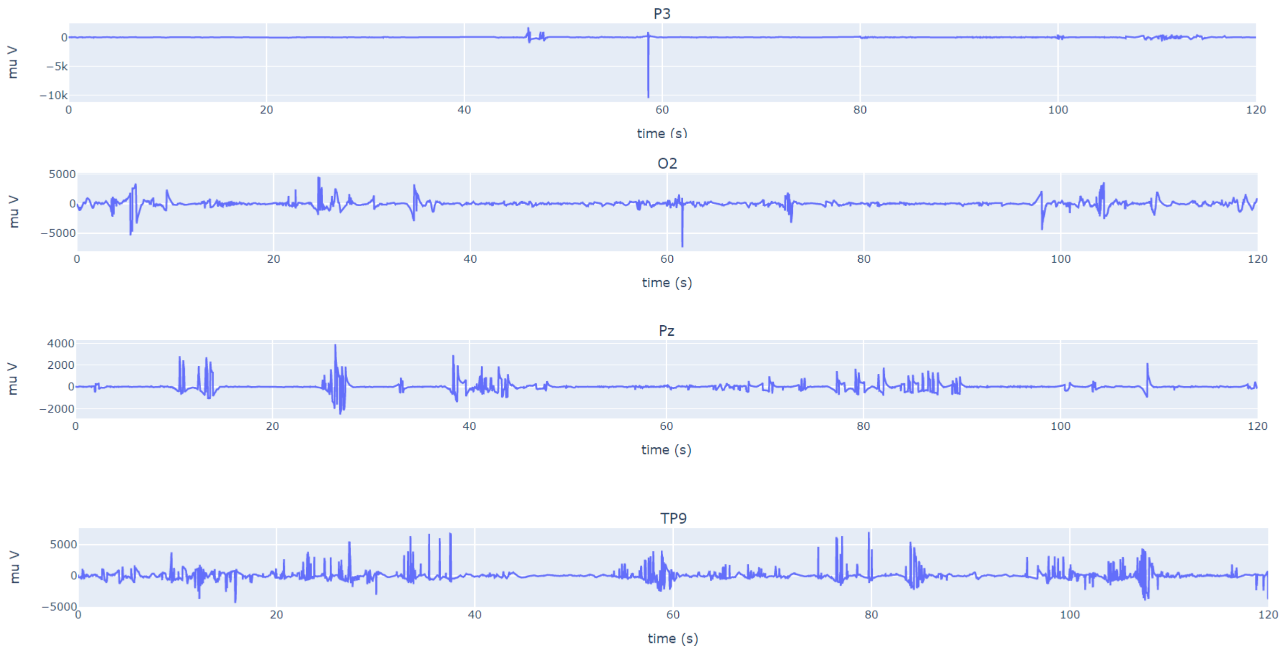


Figure 8.6.10: Four raw EEG signals from the red cluster (cluster 3)

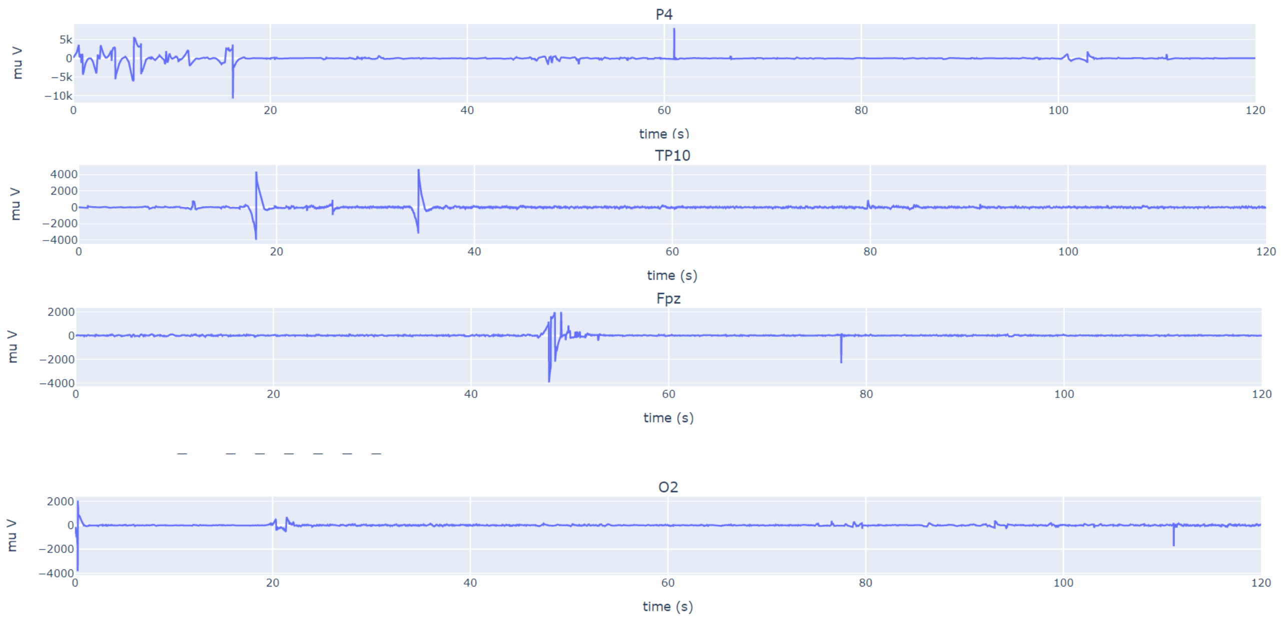


Figure 8.6.11: Four raw EEG signals from the yellow cluster (cluster 2)

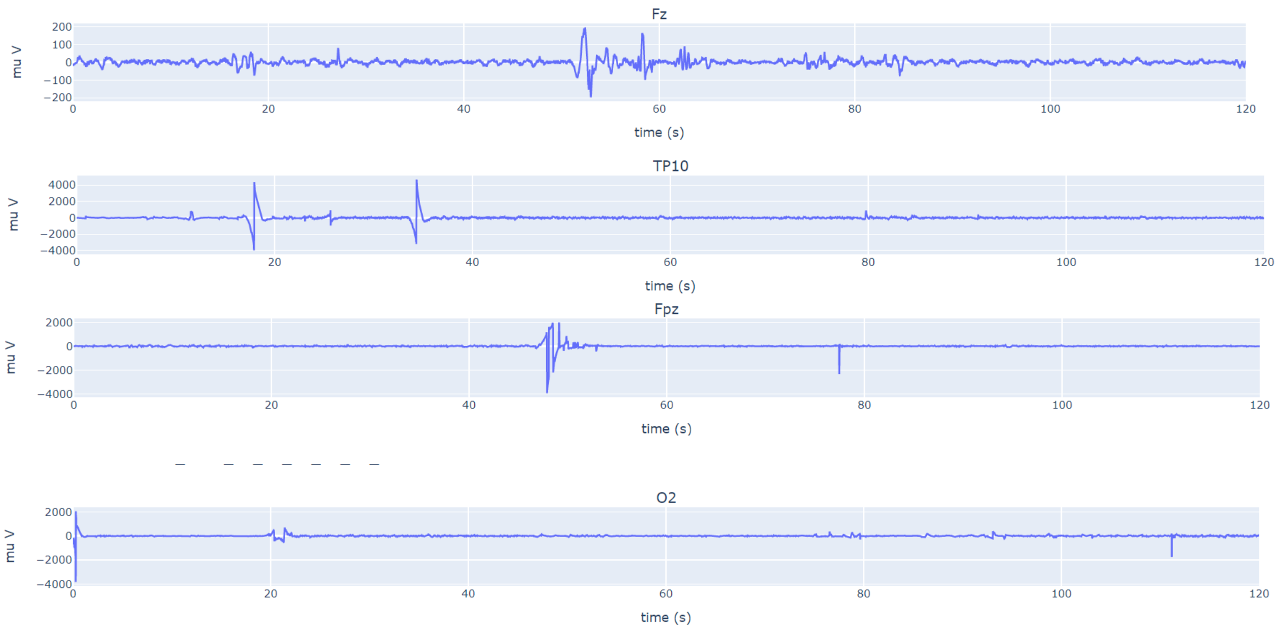


Figure 8.6.12: Four raw EEG signals from the turquoise cluster (cluster 1)

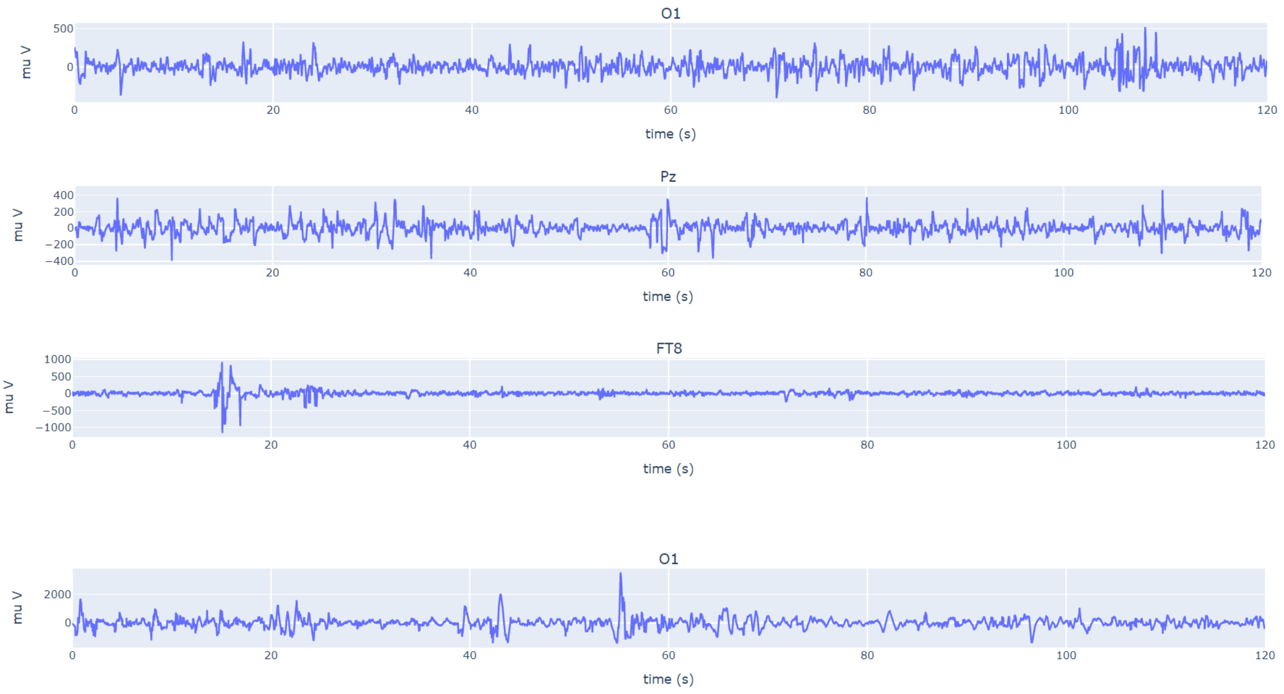


Figure 8.6.13: Four raw EEG signals from the dark blue cluster (cluster 0)

8.7 Example interactive plots

Example of how the interactive plot can be used to find the same point in PCA and t-SNE and then trace the raw data

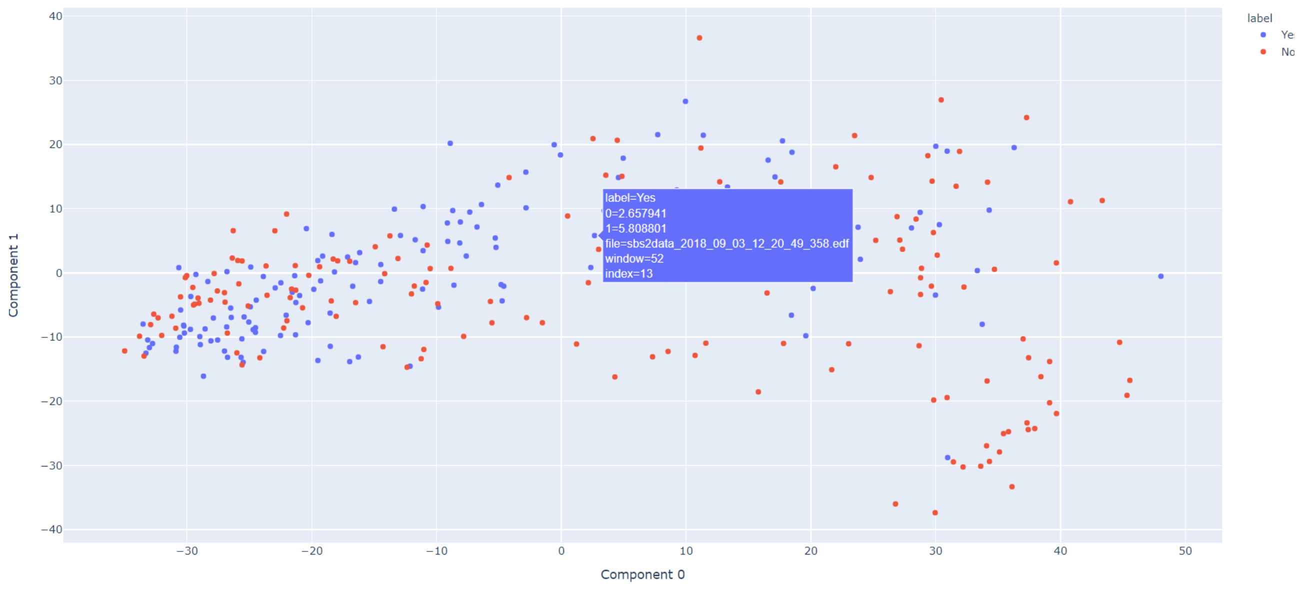


Figure 8.7.14: PCA

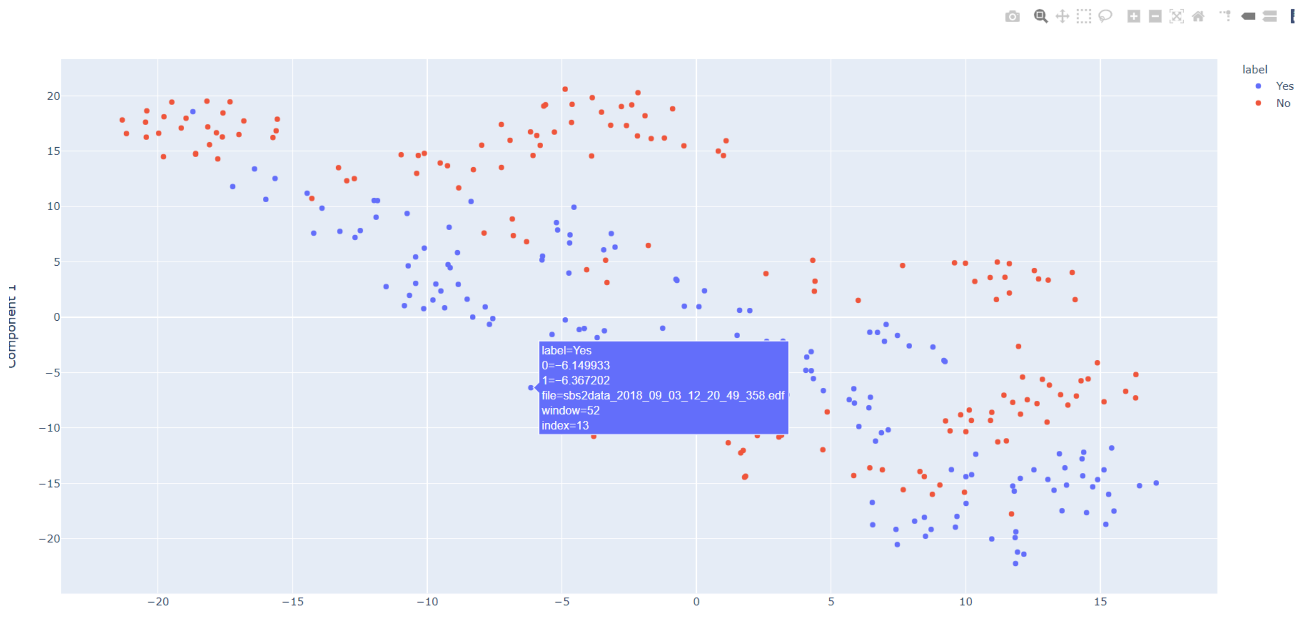


Figure 8.7.15: t-SNE

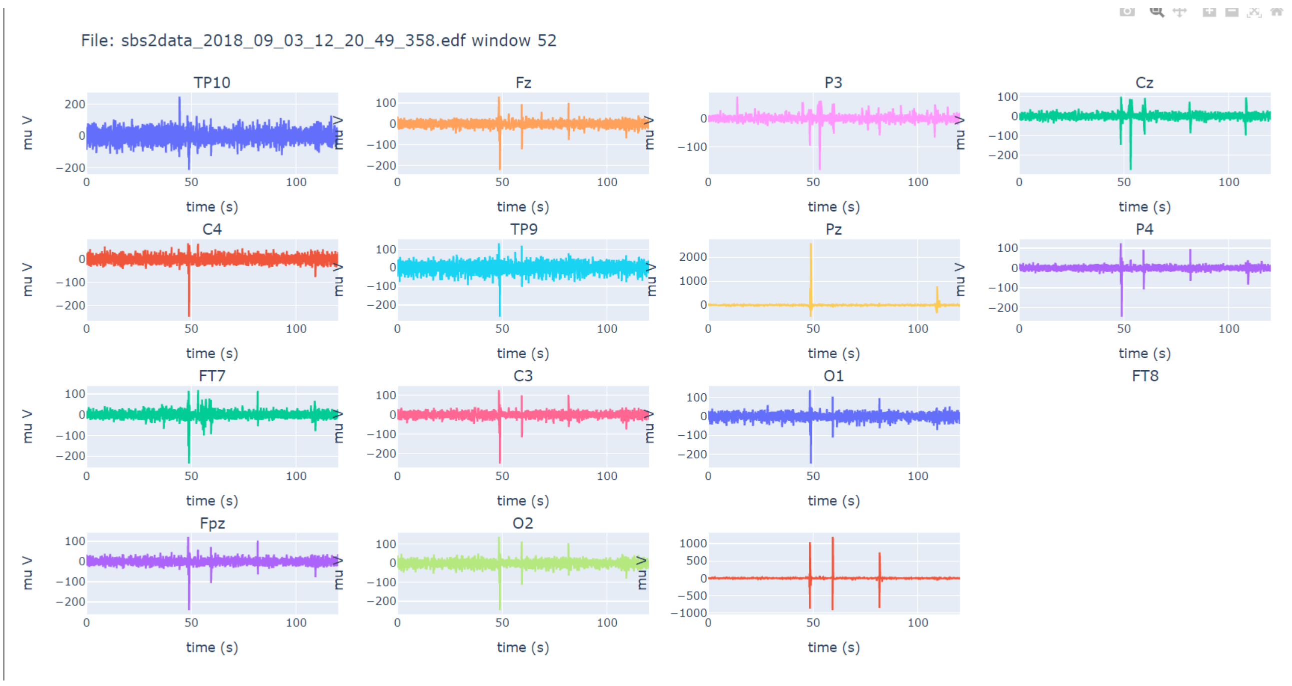


Figure 8.7.16: All channel EEG

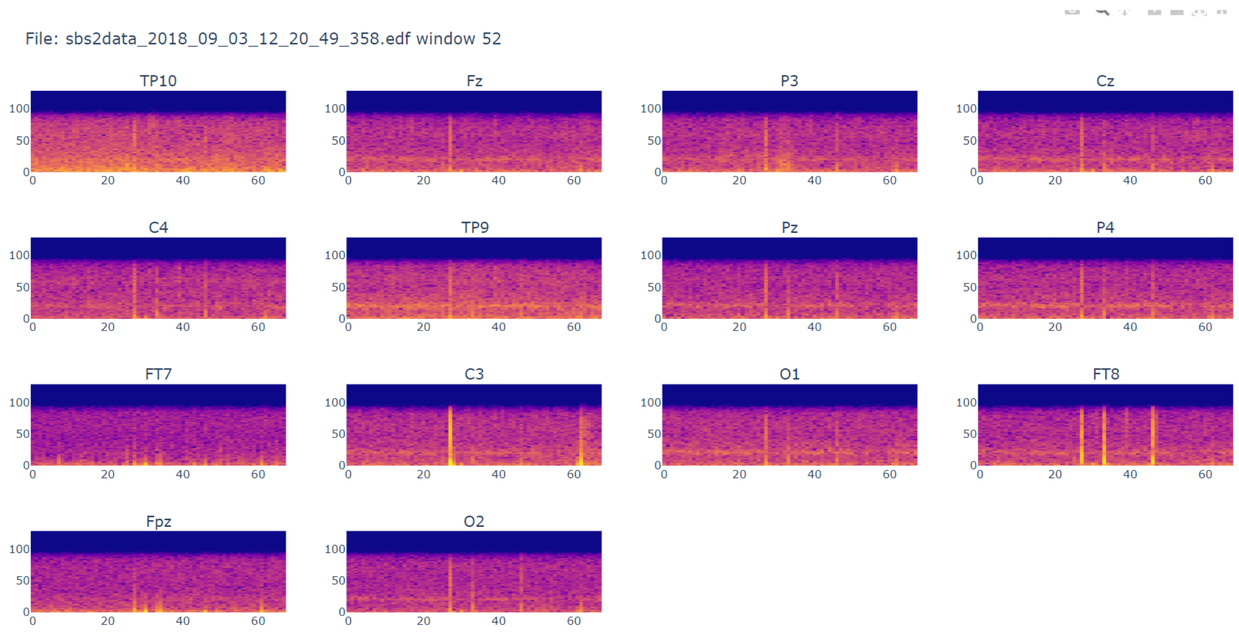


Figure 8.7.17: Spectrograms for all channels

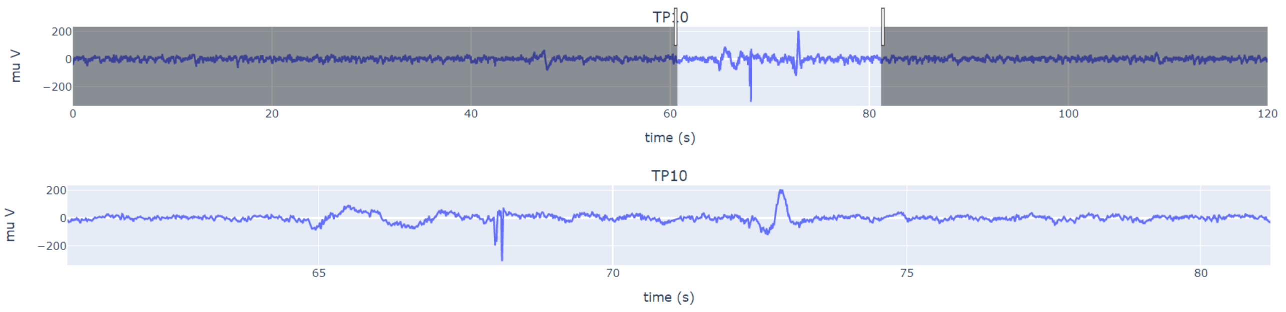


Figure 8.7.18: Example of zoom on raw EEG signal