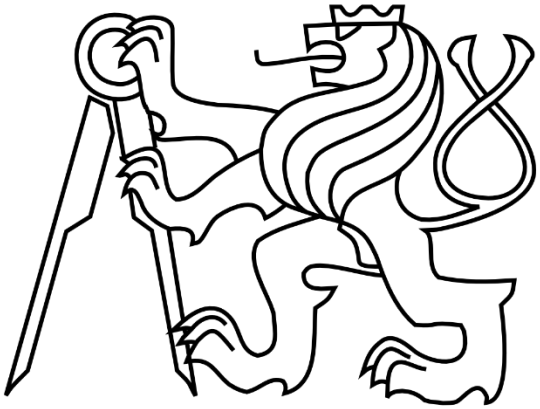
Czech Technical University
Faculty of Electrical Engineering
Department of Cybernetics



Bachelor's thesis

Artefact detection in micro-EEG signals

Tomáš Novák

Supervisor: Ing. Eduard Bakštein

Study programme: Cybernetics and Robotics

Study branch: Systems and control

May 2015

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Tomáš Novák**

Studijní program: Kybernetika a robotika
Obor: Systémy a řízení

Název tématu: **Detekce artefaktů v mikro-EEG signálech**

Pokyny pro vypracování:

1. Seznamte se s problematikou mikroelektrodových záznamů, souvisejících s hlubokou mozkovou stimulací [1].
2. Vytvořte systém pro automatickou detekci a segmentaci artefaktů pro signály získané pomocí intrakraniálních mikroelektrod.
3. Vlastnosti modelu otestujte na databázi mikroelektrodových záznamů pacientů s Parkinsonovou nemocí a porovnejte chování Vámi navrženého modelu s ruční anotací.
4. Nastudujte existující řešení [2-3] a porovnejte s nimi Vámi vytvořený systém.

Seznam odborné literatury:

[1] Israel, Z, Burchiel, KJ: Microelectrode Recordings in Movement Disorder Surgery, Thieme New York, USA, 2004, ISBN 1-58890-172-4
[2] J.H. Falkenberg, J. McNames, M. Aboy, K.J. Burchiel: Segmentation of extracellular microelectrode recordings with equal power, in proceeding of: Engineering in Medicine and Biology Society, 2003.
[3] Aboy M, Falkenberg JH.: An automatic algorithm for stationary segmentation of extracellular microelectrode recordings. Med Biol Eng Comput. 2006 Jun;44(6):511-5.

Vedoucí: Ing. Eduard Bakštein

Platnost zadání: do konce letního semestru 2015/2016

L.S.

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 7. 1. 2015

Acknowledgement

I wish to thank Eduard Bakštein for his patience and support and especially for the frequent helpful consultations and his constant will to direct me during the process.

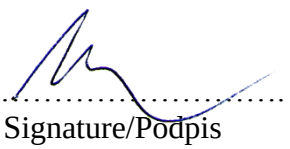
Statement

I hereby declare that this thesis is my own work and that I stated all the resources used in accordance with "Metodický pokyn o dodržování etických principů při přípravě vysokoškolských závěrečných prací"

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etnických principů při přípravě vysokoškolských závěrečných prací.

V Praze, dne 18. května 2015
In Prague, May 18th 2015



……
Signature/Podpis

Abstract

The purpose of this thesis is to design a classifier for artefact detection in microelectrode recording signals that were recorded during deep brain stimulation surgeries. Firstly, the issues of Parkinson's disease treatment, deep brain stimulation, microelectrode recording signals and existing artefact detection and segmentation methods are discussed. Secondly, two methods for artefact detection are designed. The first method is designed for power artefact detection in the time-domain using segments that are one second long. Its results are compared with the current annotation, questioning the convenience of the annotation. Then another – more general – method for artefact detection that uses machine-learning principles is suggested. Three features based on the spectrogram of the signal are designed and classifiers are created using the AdaBoost and the RUSBoost algorithms. The results are compared to a simple decision tree method and several existing methods based on the change-point detection. The final classifier based on the RUSBoost model working with one-second intervals proved capable of detecting artefacts with its 88.61% accuracy and balanced precision and specificity.

Abstrakt

Tato práce se zabývá navrhováním klasifikátorů pro detekci artefaktů v signálech nahraných mikroelektrodami v průběhu procedury hloubkové mozkové stimulace. Nejprve je nastíněna problematika léčby Parkinsonovy choroby, hloubkové mozkové stimulace, mikroelektrodových záznamů a současných způsobů detekce artefaktů a segmentace. Navrženy byly dvě metody pro detekci artefaktů. První metoda detekce je navržena pro detekci artefaktů v amplitudě signálu za použití jednovteřinových segmentů. Její výstup je porovnán se současnou anotací s výsledky zpochybňujícími vhodnost použití anotace. Následně je navržena metoda pro obecnou detekci všech pozorovaných typů artefaktů s využitím principů strojového učení. Navrženy jsou tři příznaky vycházející ze spektrogramu signálu a výsledné klasifikátory jsou vytvořeny pomocí algoritmů využívajících boosting. Výsledky jsou porovnány s jednoduchou metodou rozhodovacích stromů a existujícími metodami, založenými na detekci stacionárních oblastí. Konečný klasifikátor je založen na modelu RUSBoost. Výsledky po následné filtrování detekce na jednovteřinové intervaly jsou dostačující a klasifikátor s přesností 88.61% může být použit jako značná pomoc při odstraňování artefaktů.

Table of Contents

1 Introduction 1
2 Deep brain stimulation in Parkinson's disease 2
2.1 Parkinson's disease 2
2.2 The history of PD treatment 2
2.3 Deep brain stimulation 2
3 MER signals 4
3.1 Recording 4
3.2 Signal properties 4
3.3 The impact of artefacts on MER processing 5
3.4 MER artefacts 8
3.4.1 Power artefacts 8
3.4.2 Technical artefacts 8
3.4.3 Baseline artefacts 8
3.5 MER artefacts and existing detection methods 10
3.5.1 Manual detection 10
3.5.2 ANOVA statistical comparison 10
3.5.3 Stationary segmentation 10
4 Power artefact detection in time-domain 13
4.1 Database 13
4.2 Test and training set 13
4.3 Feature 13
4.4 Algorithm 14
4.5 Training ad-hoc constant 16
4.6 Results 17
5 Machine-learning method 20
5.1 Basic idea for this method 20
5.2 Data collection and annotation 21
5.2.1 Annotation 21
5.3 Feature extraction and selection 23
5.3.1 ROC and AUC 23
5.3.2 Feature 1: Standard deviation of frequency spectrum 23
5.3.3 Feature 2: Kolmogorov-Smirnov difference from mean frequency spectrum 24
5.3.4 Feature 3: Maximal difference from the mean frequency spectrum 26

5.3.5 Feature selection 27
5.4 Model selection 28
5.4.1 Boosting 28
5.4.2 AdaBoost 29
5.4.3 RUSBoost 30
5.4.4 Decision trees 31
5.5 Results 32
6 Conclusion 35
7 References 36
8 Appendix: CD contents 37

List of Figures

Figure 2.1: A diagram of patient with DBS 3
Figure 3.1: Power spectral density of clean signal 4
Figure 3.2: Detected spikes and their clusters 5
Figure 3.3: Spike detection demonstration 6
Figure 3.4: Spike detection in a signal with power artefact 6
Figure 3.5: A signal with artefacts and detected spikes 7
Figure 3.6: Short power artefact 8
Figure 3.7: A technical artefact lasting for the whole recording. 9
Figure 3.8: A baseline artefact in the signal. 9
Figure 4.1: ROC curves comparison 14
Figure 4.2: A detection with several different thresholds 15
Figure 4.3: The MATLAB code of the classifier 16
Figure 4.4: An Artefact detected by the classifier, but missed by the annotation 18
Figure 4.5: The technical artefact is not detected 19
Figure 4.6: A wrong annotation of the artefact 19
Figure 5.1: The design cycle of the classifier with supervised learning 20
Figure 5.2: The comparison of the annotation 21
Figure 5.3: The annotation tool 22
Figure 5.4: Feature 1 calculated from a signal with a great artefact 24
Figure 5.5: PDF and CDF of the function with normal distribution 25
Figure 5.6: Feature 2 calculated from a signal with a great artefact 26
Figure 5.7: Feature 3 calculated from a signal with a great artefact 27
Figure 5.8: ROC curves and the AUC of all features 28
Figure 5.9: The weight of a component classifier based on a training error 29
Figure 5.10: ROC of the training set 32
Figure 5.11: Results of the classification 34

List of Tables

Table 3.1: Spike frequency disparity..... 5
Table 4.1: Confusion matrix at threshold level C = 1.18 17
Table 4.2: Confusion matrix of test set 17
Table 5.1: The results of the classification of the test set..... 32
Table 5.2: The results of the classification of the test set using one-second interval 33
Table 5.3: Precision and sensitivity of the results 33

1 Introduction

Microelectrode recording (MER) is an important technique that helped neurophysiological scientists with breakthrough discoveries about the function of the nervous system. One of its important applications is the deep brain stimulation (DBS), a surgical procedure that reduces the symptoms of Parkinson's disease (PD) by delivering voltage into certain areas of the brain, e.g. subthalamic nucleus (STN). This is accomplished with microelectrodes leading from neurostimulator to the target area of the the brain. DBS is also tested to help with other chronic conditions, such as chronical pain, depression and Tourette syndrome. During the DBS surgery, MER recordings are used to distinguish between different nuclei based on their neuronal activity and also for various research purposes. The amount of researchers working with MER data is growing and all tools that help with processing extensive MER signal databases are beneficial for the future of this field of research.

This thesis is focused on offline detection of artefacts in MER signals. This detection should help researchers with the removal of contaminated segments from their database in order to improve any process performed with data. At the beginning I studied and described Parkinson's disease and its treatment (Section 2.1 and 2.2), deep brain stimulation (Section 2.3) and properties of MER signals (Chapter 3). I have studied existing solutions and pointed out their drawbacks (Section 3.5), then I suggested my own solutions. Firstly, I designed and tested a simple method using an elementary feature but an adaptive threshold (Chapter 4). Then I used a more sophisticated machine learning method that involved an annotation of 500 signals, extracting and selecting features, choice of the proper model and evaluation (Chapter 5).

The DBS toolbox for MATLAB® by Departments of Cybernetics, FEE, CTU was used for processing the database. The official Statistics and Machine Learning Toolbox was also extensively used during the whole process.

2 Deep brain stimulation in Parkinson's disease

In human bodies, MER signals are recorded during invasive brain surgeries, thus there are not many possibilities to obtain this data. Recordings from DBS surgeries for patients with Parkinson's disease are the most frequent source of MER signals.

2.1 Parkinson's disease

Parkinson's disease is an incurable movement disorder characterized by rigidity, tremor, bradykinesia and akinesia [1]. The prevalence of this most common serious disorder in the world is 0.3% in industrialized countries. The disorder affects 1% of people who are older than 60 years and 4% of highest age groups [2].

Cells forming basal ganglia have great influence on motoric skills. They communicate using a chemical compound called dopamine, which is produced in the part of brain that is called substancia nigra. The basal ganglia works improperly if neurons are dying in the substancia nigra which causes the lack of dopamine [3].

2.2 The history of PD treatment

The existence of electricity inside living organisms is known since the 18th century. The brief electric discharge - action potential - was discovered in the 19th century by Emil du Bois-Reymond and Ludimar Hermann. It took some time before a sufficient measuring device was invented. In 1921 Ida Henrietta Hyde invented the intracellular microelectrode. This led to many important discoveries about the nervous system and it generally shifted further studies of neurophysiology.

MER became widely used during stereotactic movement disorder surgeries. In 1968, the standard L-DOPA (chemical) therapy for PD was introduced and replaced stereotactic surgery in late-stage patients. This therapy delivers lacking dopamine into the brain. However L-DOPA therapy has several drawbacks. The effect weakens during long-time usage, thus higher doses administrated more frequently are necessary. L-DOPA can also cause side-effects among which sickness and mental problems are listed.

In 1982 several drug addicts used heroin batches contaminated with MPTP, 1-methyl-4-phenyl-1,2,3,6-tetrahydropyridine. This neurotoxin was synthesized in 1947 as an analgesic. All the drug addicts developed main symptoms of PD. This was the beginning of a great development in PD research. Scientists could study the disease on primates administered with MPTP. In the late 1980s, STN was identified as a potential therapeutic target for the assuagement of PD symptoms [4].

2.3 Deep brain stimulation

The deep brain stimulation is a surgical procedure for PD patients. It is done by the insertion of a permanent electrode to the target nucleus. This electrode is connected to a pacemaker and stimulates the targeted area. To precisely select the trajectory for the electrode, magnetic resonance imaging is done before the operation. During the DBS surgery, the surgeon is visually evaluating MER signals by examining its time-domain and listening to the signal to determine the accurate location of the microelectrode.

DBS is very successful in assuagement of PD symptoms. It is used mostly for patients who do not respond to chemical therapy anymore. Patients have to visit their doctors periodically after the surgery so that the doctor can correct the pacemaker's settings. Amplitude, length and frequency of stimulating pulses can be adjusted. Figure 2.1 shows diagram of patient with DBS.

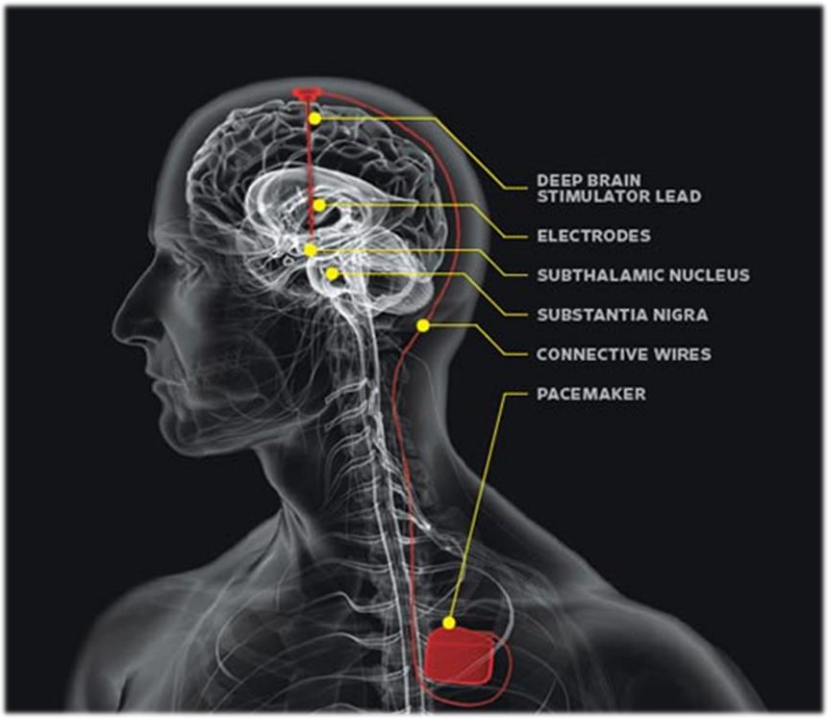


FIGURE 2.1: A DIAGRAM OF PATIENT WITH DBS

Patient after DBS surgery with implanted electrodes and a pacemaker for settings and control [5].

3 MER signals

This chapter describes properties of MER signals and their recording. It also discuss different origin and types of artefacts that occurs during recording and current methods of said contaminated signal removal.

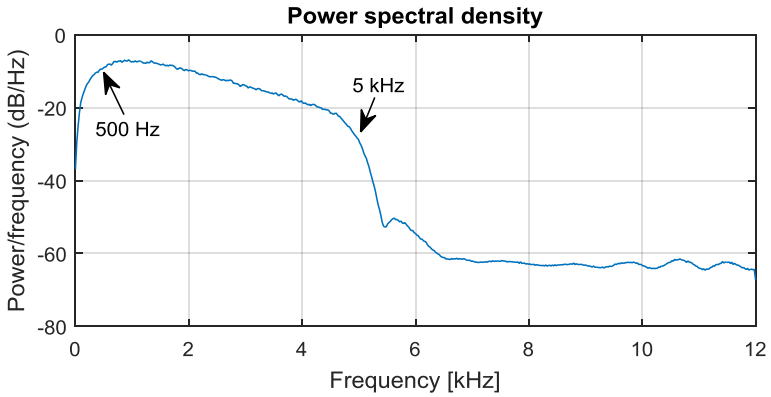


FIGURE 3.1: POWER SPECTRAL DENSITY OF CLEAN SIGNAL

Effects of band-pass filter are visible although the change is gradual. The maximum frequency is only a half of the sampling frequency because the frequency spectrum of real function is symmetrical.

3.1 Recording

MER signals are recorded during the DBS surgery by microelectrodes with a tip diameter of about 1-10 μm. Most suitable materials for microelectrodes are tungsten and platinum-iridium alloy. Microelectrode wire follows the calculated trajectory through the brain and stops at several positions where the signal is recorded. Most of the recordings use from 1 to 5 channels (in parallel inserted electrodes). Our data is recorded using the Leadpoint recording system by Medtronic, sampled with 24 kHz and filtered with 500 Hz to 5 kHz band-pass filter during the recording. The effect of this filter can be seen in Figure 3.1. With this filter, maximum frequency f_{max} of signal is 5 kHz (upper cutoff frequency). According to sampling theorem, perfect reconstruction of signal is possible if

$$f_s > 2f_{max}, \tag{1}$$

where f_s is the sampling frequency. Our data meets this condition by a wide margin. The data is oversampled by factor of 2.4, because it is 2.4 times higher than the Nyquist rate ($2f_{max}$).

3.2 Signal properties

MER recordings are also called μEEG, but they are different from EEG signals. Unlike EEG, MER signals are not recorded on the surface of the head, but directly inside the brain. The diameter of microelectrode's tip is comparable to the size of the neuron which allows the microelectrode to accurately record the activity of several closely positioned neurons.

The MER signal consist of two main components: background (noise) and spikes (action potentials). In most studies, background is considered to be an unwanted element generated by the activity of distant neurons. However, for other purposes – such as nuclei identification in

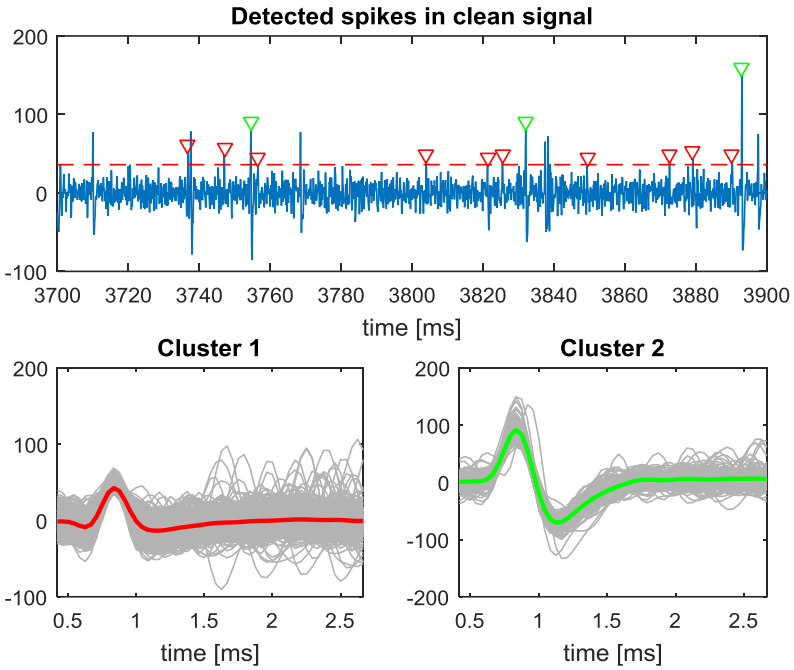


FIGURE 3.2: DETECTED SPIKES AND THEIR CLUSTERS

Short stretch of a signal recorded in STN, detected and clustered spikes. Each cluster is generated by a different neuron.

PD – background activity may be an important and investigated signal property. The spike is an event in which the electrical membrane potential of a neuron rapidly grows and decreases. Several neurons can contribute on the spike activity in the signal, but their position relative to the electrode defines the shape and the amplitude of the spike, thus they can be divided into clusters. In Figure 3.2, we can see two cluster of spikes detected by the spike sorting algorithm from clean signal recorded in STN and their mean. In the figure one can also see a part of this signal with red arrows pointing at detected spikes from cluster 1 and green arrows pointing at spikes from cluster 2. The red line represents the threshold evaluated by the algorithm – everything exceeding this threshold is considered a spike.

3.3 The impact of artefacts on MER processing

As mentioned in Section 2.3, the surgeon determines the accurate location of the microelectrode during the DBS surgery. There is a research working with enormous amount of raw data that is trying to automatize this decision. To do this, they need to design a classifier using data features (e.g. average power). However, raw data can contain a large number of artefacts that make valid feature extraction hard or even impossible. This problem does not apply to brain navigation only but also to many other researches in which scientists work with MER data.

t=(0,5)∪(6,10) [Hz]	t=(0,10) [Hz]
34.6	43.5

TABLE 3.1: SPIKE FREQUENCY DISPARITY

Detected changes in neuronal mean firing rate if we ignore/include part of the signal with artefact.

One of the basic operations with MRE data is the spike sorting, a procedure that tries to separate

action potentials (spikes) from background noise. See Figure 3.3. As we can see in Figure 3.4 and Figure 3.5, the power artefact affects the spike detection algorithm and introduces an error in its results. For example, using this signal to determine spike frequency can cause major inaccuracy. See Table 3.1.

Detecting and labelling or filtering artefacts in MER signals is a complex task. An algorithm [7][8] that should be able to select stationary (clear) parts of the signal has already been developed, but it has some problems that are discussed in Section 3.5.3.

The goal of this thesis is to design a classifier for automatic detection of artefacts that could help anyone who is working with MER data to effectively identify raw data that are applicable for further use from unusable data in the sense of quality. This classification should be precise enough to label most of the basic artefacts without the need for more detailed distinction of artefact types. There should also be as few false positive classifications as possible to preserve most parts of clean signals.

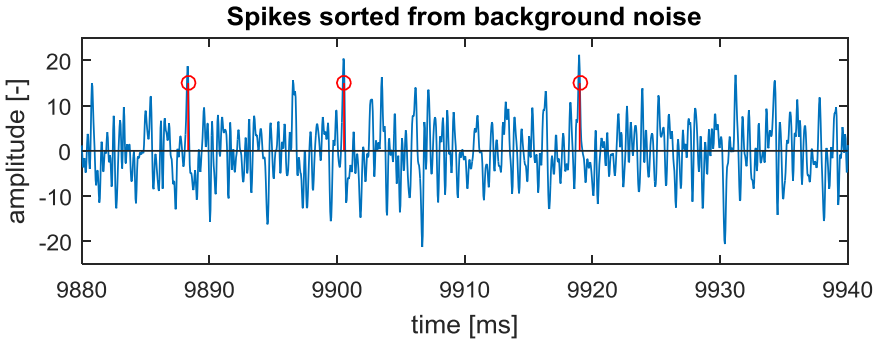


FIGURE 3.3: SPIKE DETECTION DEMONSTRATION

The red stems with a circle at the top represents detected spikes.

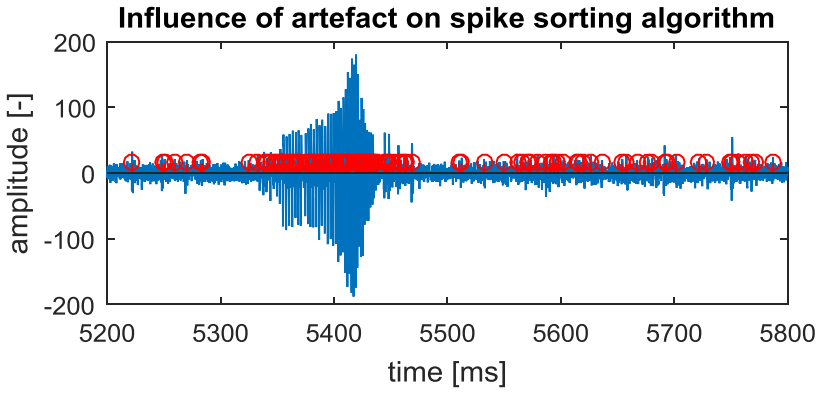


FIGURE 3.4: SPIKE DETECTION IN A SIGNAL WITH POWER ARTEFACT

Artefacts clearly affect the spike detection. The frequency of spikes in the short segment where the artefact occurs is greatly increased.

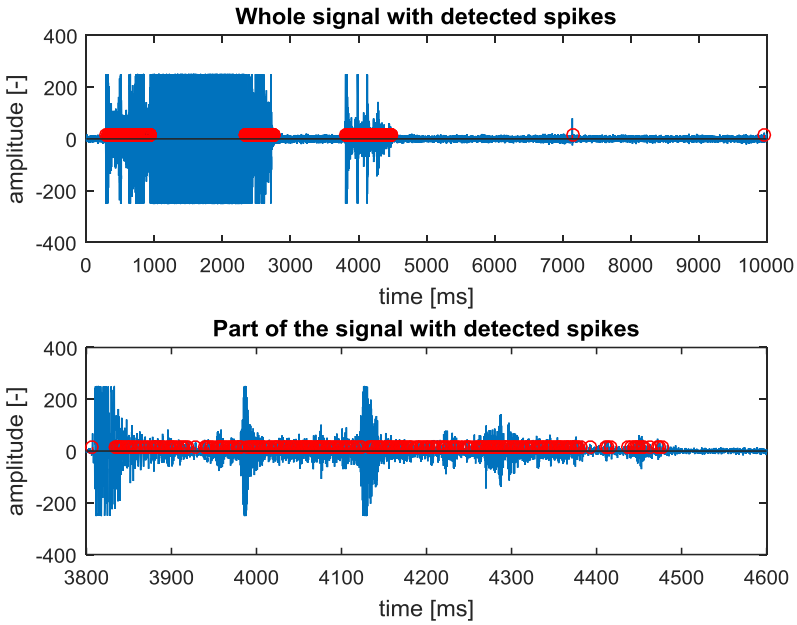


FIGURE 3.5: A SIGNAL WITH ARTEFACTS AND DETECTED SPIKES

The algorithm only detected spikes when artefacts occurred. Artefacts affected this spike detection highly.

3.4 MER artefacts

When inspecting the signals in our database, we encounter artefacts varying in both temporal and spectral properties. For the evaluation of a detection system, it may be useful to divide the artefacts into several groups based on their properties. This way we can identify weaknesses of our algorithms and have an overview of how frequent different artefacts are in the data. According to the catalogue of artefact types that was made for the research at CTU FEE there are 5 types of artefacts. I only used 3 categories in this thesis because I found out that they often overlap. Also, it is not as critical to recognize the type of an artefact as to find it.

3.4.1 Power artefacts

This type of artefact is described as a change in the signal amplitude. Typically they are about 100 milliseconds to 2 seconds long and the amplitude change is large, occasionally clipping out of the range. Using a spectrogram, they can be easily spotted as an amplification in the wide frequency spectrum as can be seen in Figure 3.6.

Sources of these artefacts are mainly the patient's movements during the recording. They are very common in MER and have great influence on most signal features. Having them labelled and excluded is necessary.

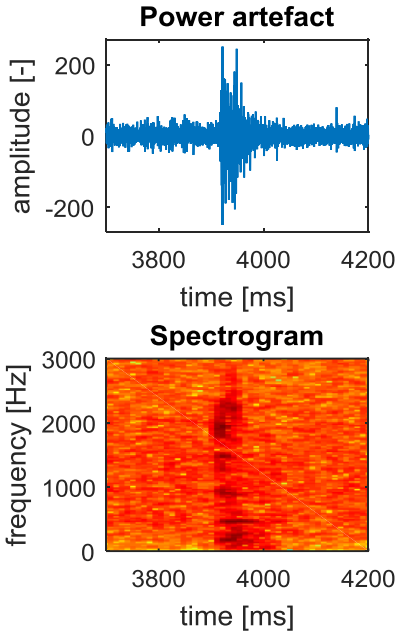


FIGURE 3.6: SHORT POWER ARTEFACT

A very short power artefact and its effect on the spectrogram – a narrow stain in a wide frequency range.

3.4.2 Technical artefacts

Technical or frequency artefacts are represented by a long-term constant frequency in the signal's frequency spectrum, thus they cannot be physiological. They might not be easily spotted in the time-domain, especially if the artefact lasts for the whole recording, but they are easily identified in the spectrogram as long amplifications of certain frequency in time. This can be seen in Figure 3.7. In this figure we can also see that there is no significant effect on the amplitude of signal apart from the 8th second where the artefact intensifies.

Technical artefacts mostly originate from the electromagnetic interference of other equipment such as lights and motors in the surgery room

3.4.3 Baseline artefacts

Baseline artefacts are the results of interferences of low frequencies that cause fluctuation in the baseline of the signal. They can be spotted as fluctuations in the detailed time-domain view of the signal and in its spectrogram as an amplification at the lowest frequencies. See Figure 3.8. Filters mentioned in Section 3.1 should eliminate these low frequency artefacts, but

the high-pass filter is very gradual and it is possible that very strong amplitudes retain in the filtered signal.

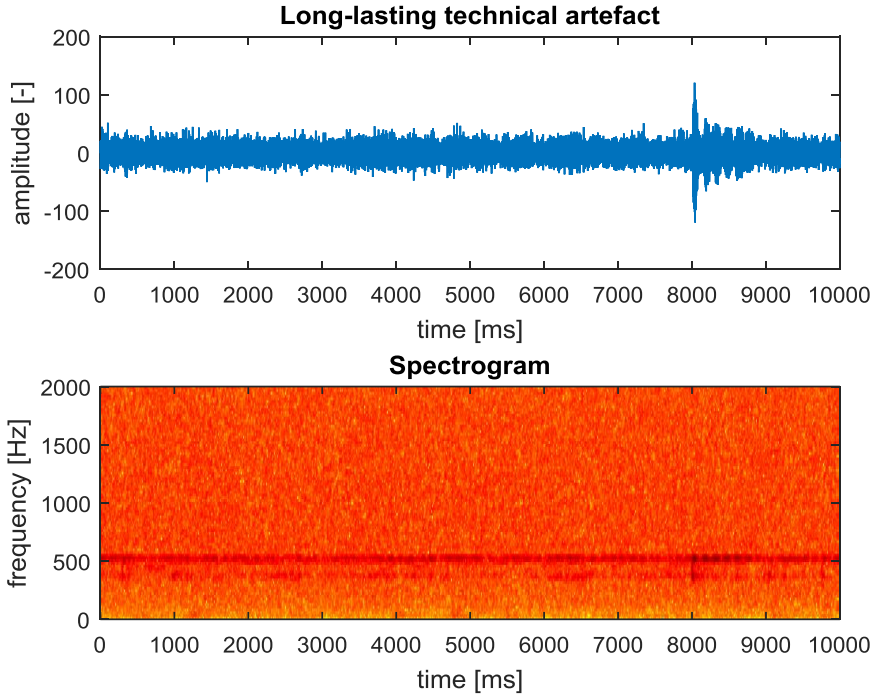


FIGURE 3.7: A TECHNICAL ARTEFACT LASTING FOR THE WHOLE RECORDING.

A technical artefact at frequency around 500 Hz. This artefact is very strong and has constant power during the whole recording.

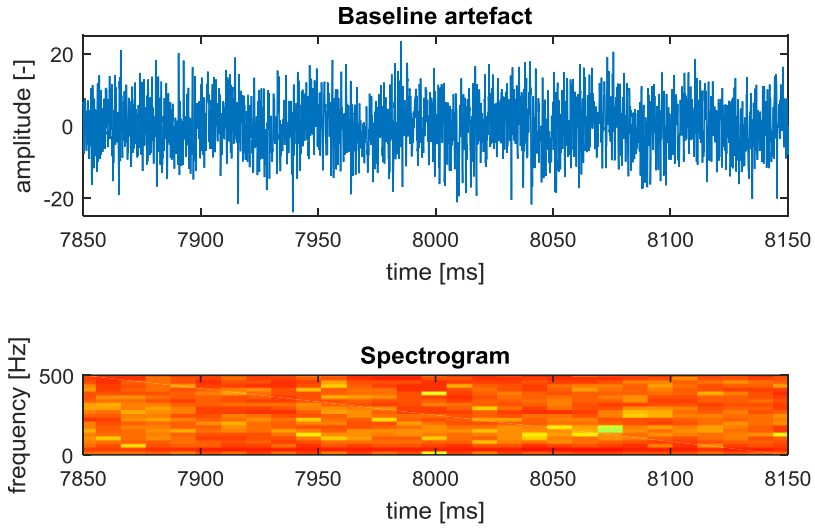


FIGURE 3.8: A BASELINE ARTEFACT IN THE SIGNAL.

The fluctuation of the signal's baseline is easy to observe. A dark stain of low frequency and long period can be observed in the spectrogram.

3.5 MER artefacts and existing detection methods

During the recording, some phenomena, either of an electrical or biological origin, can interfere with the signal thus decreasing its signal-to-noise ratio. These interferences can be reduced with proper preparation and common pre-processing. Decent grounding of the equipment in the operating room can eliminate common ambient electrical noise. Other common artefacts are caused by the movement of brain in the cerebrospinal fluid which displaces neurons relative to the electrode. This movement is caused by respiration, pulse or the movement of other body parts that causes changes in the blood circulation and possibly in the intracranial pressure [4].

Despite all the precautions and care, artefacts are common in MER signals. As shown in Section 3.3, removing artefacts from signals is important in basic operations performed with MER data. Several methods are currently used.

3.5.1 Manual detection

The simplest and the most often used method is manual detection performed by trained evaluators. Generally this method can lead to good results, but it is highly dependent on the evaluator's experience and the quality of the software. A visual inspection of signals at different levels of detail can reveal artefacts highly affecting waveform of the signal, but long-lasting artefacts can be easily overlooked. If a spectrogram, PSD or other visual representation is added the recognisability of artefacts is improved, but the evaluator's interaction with the visualisation is increased thus time demands are raised as well. This can be a great drawback for vast databases like ours. The reliability of this method can be increased by the duplicity of detection of the same data, the comparison of the results among evaluators and the construction of the final database with the use of a decision based on the majority vote or something similar.

3.5.2 ANOVA statistical comparison

A simple algorithm for the detection of signals contaminated with artefacts is used in [6]. The authors reject the whole recording based on two conditions. Firstly, signals with an amplitude greater than the threshold of 300 μV are rejected. Secondly, root mean square (RMS) values of non-overlapping 20 ms windows for the first and the last two seconds of the signal are calculated. Then ANOVA (analysis of variance) is used to determine whether RMS values of the first and the last two seconds come from the same distribution at the statistical significance level 0.005. This method can be successful especially in their study that uses mainly RMS values, but is still unable to detect long-lasting artefacts or artefacts occurring between the first and the last two seconds of the signal. Also rejecting the whole signal may be an unnecessary waste of the signal in the case of short artefacts that are also common in MER signals.

3.5.3 Stationary segmentation

A more advanced method was developed by Aboy and Falkenberg in [7][8]. MER data is used during the DBS surgery as a reliable tool for the surgeon. It helps him to accurately determine the position of the microelectrode. It is believed that 3 second is the minimum length needed to accurately determine the position. The algorithm is designed to find the longest stationary segment in the signal to help the surgeon with his decisions.

Standard score (normalization) is evaluated from signal **x** as

$$\mathbf{y} = \frac{x-\mu_x}{\sigma_x}, \tag{2}$$

where μ_x is the mean of **x** and σ_x is the standard deviation of **x**. Then the signal is divided into *N* non-overlapping segments **s** with equal length t,

$$\mathbf{s} = \{s_1, s_2, \dots, s_N\}. \tag{3}$$

Autocorrelation function *r* is evaluated for every segment of vector **s** forming vector **a** as

$$a_i = r(\mathrm{s}_i), i \in \langle 1, N \rangle, \tag{4}$$

$$\mathbf{a} = \{a_1, a_2, \dots, a_N\}. \tag{5}$$

Variance of every autocorrelation function is evaluated into vector $\boldsymbol{v}$ as

$$v_i = var(a_i), i \in \langle 1, N \rangle, \tag{6}$$

$$\boldsymbol{v} = \{v_1, v_2, \dots, v_N\}. \tag{7}$$

From the vector of variances $\boldsymbol{v}$, the algorithm calculates the vector of ratios (distances) $\boldsymbol{r}$ of all adjoining elements as

$$r_i = \frac{max(v_i, v_{i+1})}{min(v_i, v_{i+1})}, i \in \langle 1, N-1 \rangle, \tag{8}$$

$$\boldsymbol{r} = \{v_1, v_2, \dots, v_{N-1}\}, \tag{9}$$

where the nominator is always the higher value and the denominator is always the smaller value.

The vector $\boldsymbol{r}$ represents the distance between neighbouring segments. When this distance is higher than a certain threshold, time boundary between those neighbouring segments is considered to be a transition. A part of the signal with the greatest difference between the start and the end of this part is the longest stationary segment and is returned as the result.

My colleagues from the Neuroscience research group at the Department of Cybernetics further extended this algorithm to find a non-contiguous stationary segment in the signal which reduces the wastage of clean parts of the signal discarded by the original version. It is more suitable for the comparison with a manual annotation. The ratio of variances (distance) is evaluated for every possible pair, not only for the adjoining elements, forming a distance matrix *R* as

$$r_{i,j} = \frac{max(v_i, v_j)}{min(v_i, v_j)}, i \in \langle 1, N-1 \rangle, j \in \langle 1, N-1 \rangle, \tag{10}$$

$$R = \begin{pmatrix} 0 & r_{1,2} & \cdots & r_{1,N-1} \\ r_{2,1} & 0 & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ r_{N-1,1} & r_{N-1,2} & \cdots & 0 \end{pmatrix}. \tag{11}$$

Values of R higher than the threshold are replaced by ones and other values are replaced by zeros. Greedy algorithm searching for longest sequence of zeros is applied in graph created

from transition matrix R – 0 is a transition, 1 is not a transition. The outcome of this process is represented by the indexes of the segment that does not have to be contiguous.

The main drawback of these methods is that they do not use prior information about the data other than the search of an ad-hoc threshold. As a result, if a long-lasting stationary artefact is present, it might be the longest stationary segment and thus it would be returned as the result. Another problem are signals with large variability such as signals recorded in STN. Its variability causes false transitions and greatly shortens the final segment although the signal is clean.

4 Power artefact detection in time-domain

The first out of the two designed artefact classifiers is focused on power artefacts detection, i.e. the changes of amplitude in time-domain. This method is based solely on the time course of MER signals and it uses an iteratively adaptive threshold. The results were not sufficient, therefore I developed another, more sophisticated method in Chapter 5.

4.1 Database

Our actual database consist of 18384 MER signals from 71 patients with the total length of almost 50 hours. The standard length of signals is 10 seconds but there is a small percentage of shorter signals. The term "exploration" means one trajectory performed by electrodes and "positions" are the places where the electrode stops for recording. The medical nomenclature distinguishes several types of electrodes: central, anterior, posterior, lateral, and medial according to their position in the brain. All these signals were obtained in the Na Homolce Hospital in collaboration with the Department of Neurology, 1st Faculty of Medicine, Charles University in Prague. The database is structured as follows:

- Patients – 71 patients
- Explorations – 0 to 4 explorations, mean 1.9
- Positions – 18 to 62 positions, mean 36.7
- Electrodes – 1 to 5 electrodes, mean 3.7

Every signal has its own ID structured [patient id]t[exploration id]p[position id][first letter of electrode name]. For historical reasons and its backwards compatibility the exploration is not from 1 to 4 as expected, it is rather an exploratory id. The information about the nucleus was annotated by the neurophysiologist during the surgery and thus it is now available for each signal.

Artefacts of about 31% of signals were annotated manually by 5 trained evaluators from the Department of Cybernetics, FEE, CTU. They judged signals by the visual inspection of the time-domain and the spectrogram and by listening to the signal trough headphones. The annotation was divided into 5 categories according to the type of the one-second segments.

4.2 Test and training set

The test and the training set were selected from our signal database. 100 training and 50 test signals were selected randomly, except for the distribution of nuclei where the signals were recorded. The distribution was defined to fit the nuclei distribution of the whole database. The artefact annotation was also used from the database but it was filtered to only contain power artefacts, thus other artefacts were considered clean signals. Also, one special dataset where technical artefacts were not filtered was created for comparison in Figure 4.1.

4.3 Feature

The feature used for the detection is simply a standard deviation (STD) of signal segment. The STD is a root mean square deviation from mean [9], computed according to

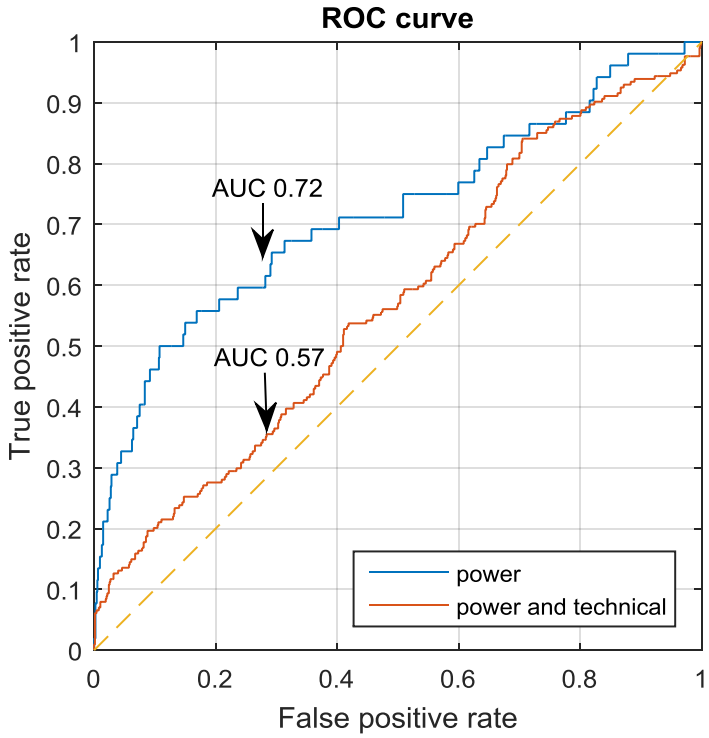


FIGURE 4.1: ROC CURVES COMPARISON

A comparison of several ROC curves of the STD feature. It is clear that the feature is better for detecting the power artefacts.

$$y = STD(\boldsymbol{x}) = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}, \tag{12}$$

where $\boldsymbol{x}$ is the signal, $\bar{x}$ is the mean of the signal $\boldsymbol{x}$ and N is the number of samples of the signal $\boldsymbol{x}$.

For its properties this feature is good for detecting power artefacts assuming sufficiently long intervals from which it is calculated. However, it is not very suitable for the detection of technical artefacts which do not have a great impact on the amplitude of the signal. This can be observed in Figure 4.1. For this figure, feature vector of every signal was normalized by dividing it by its own sum to become comparable between signals with different power. This was necessary because unlike my algorithm, ROC uses a constant threshold for the whole dataset.

4.4 Algorithm

The idea for this algorithm came from a simple observation of the signal which showed that there can be power artefacts of different sizes in one signal. The algorithm simply repeats the threshold evaluation until no artefact is detected.

Firstly, the signal is divided into N segments of the same length forming the vector $\boldsymbol{x}$. The feature (STD) is evaluated for every segment forming the vector $\mathbf{s}$ as

$$\boldsymbol{x} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}, \tag{13}$$

$$\mathrm{s}_i = STD(\boldsymbol{x}_i), i \in \langle 1, N \rangle, \tag{14}$$

$$\mathbf{s} = \{s_1, s_2, \dots, s_N\}. \tag{15}$$

The vector of classifications $\boldsymbol{r}$ for every segment is generated and filled with zeros,

$$\boldsymbol{r} = \{0,0, \dots,0\}, size\ of\ \boldsymbol{r} = N. \tag{16}$$

Then the threshold T is evaluated as

$$T = std(\mathbf{x}) \times C, \tag{17}$$

where C is an ad-hoc constant evaluated in Section 4.5.

Every segment with a feature s_i higher than the threshold T is considered an artefact and removed from the vector **s**. The elements of the vector $\boldsymbol{r}$ with the same indexes as segments considered to be artefacts are changed from 0 to 1. If no artefact is found, the algorithm ends. If an artefact is found, the algorithm goes back to the threshold T calculation which is, as before, evaluated from the vector **s** but without segments labelled as artefacts. The result of this algorithm is a logical vector where 1 stands for the artefact in the segment and 0 stands for the clean segment. The MATLAB code of this detection can be seen in Figure 4.3.

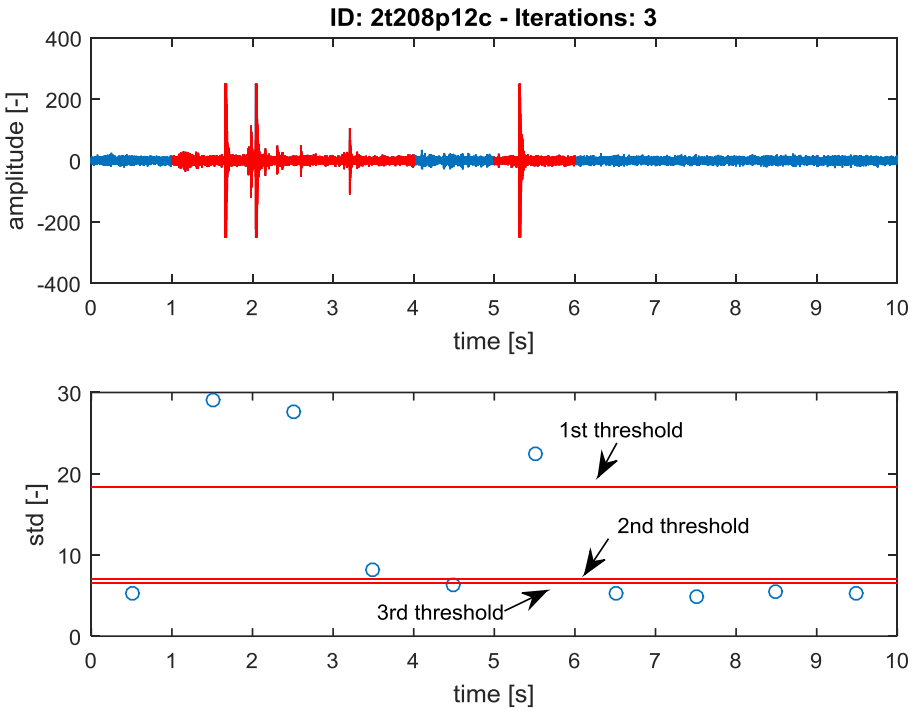


FIGURE 4.2: A DETECTION WITH SEVERAL DIFFERENT THRESHOLDS

An example of a successful detection with 3 thresholds. The first threshold detected the highest artefacts, the second threshold detected a smaller artefact and the last threshold detected nothing and the algorithm ended.

```
% Data - signal
% Interval - number of wanted intervals (recommended 10)
% C - Ad-Hoc constant (recommended 1.18)
% create the vector for the features
s=zeros(1,intervals);
% length of every segment
intSamples=int32(length(data)/intervals);
for i=1:intervals
    % divide the signal into segments
    x(i,:)=data((i-1)*intSamples+1:i*intSamples-intervals);
    % evaluate the feature for every segment
    s(i)=std(x(i,:));
end
% create the vector for classifications
r=zeros(1,intervals);
newArtif=1;
% while new artefact is found
while(newArtif==1)
    % evaluate the threshold from the vector x
    T=nanstd(reshape(x, [1 size(x,1)*size(x,2)]))*C;
    newArtif=0;
    % go through all segments
    for i=1:intervals
        % ignore the segments with an artefact
        if(r(i)==1)
            continue;
        end
        % if segments STD is higher than the threshold
        if(s(i)>T)
            % label as an artefact
            r(i)=1;
            % new artefact was found
            newArtif=1;
            % ignore this segment next iteration
            x(i,:)=NaN;
        end
    end
end
```

FIGURE 4.3: THE MATLAB CODE OF THE CLASSIFIER

The code of power artefact detection with a commentary.

One instance of detection can be observed in Figure 4.2. The first iteration detected 3 segments with a very significant power artefact. After leaving out the segments with detected artefacts, the second threshold was evaluated and detected a 4th segment with an artefact. The third threshold was above the STD of all remaining segments, thus they were considered clear.

4.5 Training ad-hoc constant

To fit with our annotation, one-second segments were used during the training, thus $N = 10$. The threshold T is calculated from the STD of the whole vector $\mathbf{s}$ multiplied by the constant C which had to be higher than 1 because we are trying to find segments with a relatively higher STD than the STD of other segments. For example if $C = 1.4$ we say that every segment with STD 1.4 times or more the whole signal's STD is an artefact.

To find the ideal value of this constant the training set was periodically classified with the values of C from 1.01 to 2 with 0.01 step. The confusion matrix, the precision, the sensitivity and the F1 score were evaluated as

$$confusion\ matrix = \begin{pmatrix} true\ positive & false\ positive \\ false\ negative & true\ negative \end{pmatrix}, \tag{18}$$

$$precision = \frac{true\ positive}{true\ positive + false\ positive}, \tag{19}$$

$$sensitivity = \frac{true\ positive}{true\ positive + false\ negative}, \tag{20}$$

$$F1\ score = 2 \times \frac{precision \times sensitivity}{precision + sensitivity}. \tag{21}$$

The maximum $F1\ score$ of 0.319 was reached for constant $C = 1.18$. Its confusion matrix can be seen in Table 4.1.

	Artefact	Clean signal
Classified as artefact	15 (36%)	27 (64%)
Classified as clean signal	37 (4%)	921 (96%)
Accuracy 93.6%	Sensitivity 29%	Specificity 97%

TABLE 4.1: CONFUSION MATRIX AT THRESHOLD LEVEL C = 1.18

4.6 Results

After fitting the constant on the training data we evaluated the classifier performance on the test set. The test set's results were better than the results obtained during the training as can be seen in Table 4.2.

	Artefact	Clean signal
Classified as artefact	20 (51%)	19 (49%)
Classified as clean signal	9 (2%)	452 (98%)
Accuracy 94.4%	Sensitivity 69%	Specificity 96%

TABLE 4.2: CONFUSION MATRIX OF TEST SET

The results do not look well if we only focus on the numbers. There are not many cases in which the classifier labelled the signal as clean even though the annotation indicated there was an artefact. However, there are many cases where the signal was labelled as an artefact by the classifier but not by the annotation.

If we look at the plots of all detections compared to the annotation, we find that the problem might be in the annotation. The annotation often misses an artefact that was detected by my classifier (Figure 4.4). I also encountered segments that do not meet the definition of a power artefact but were still considered artefacts in the annotation (Figure 4.6).

To conclude, the classifier is not able to fit on the training data adequately which resulted in a high bias. However, the visual inspection of the signals shows much better results than the ones that can be seen in the numbers. This might be because of the poor annotation of the training and testing datasets. The classifier is designed only to find power artefacts, thus long technical artefacts and other artefacts that do not affect the amplitude of the signal are not detected as seen in Figure 4.5.

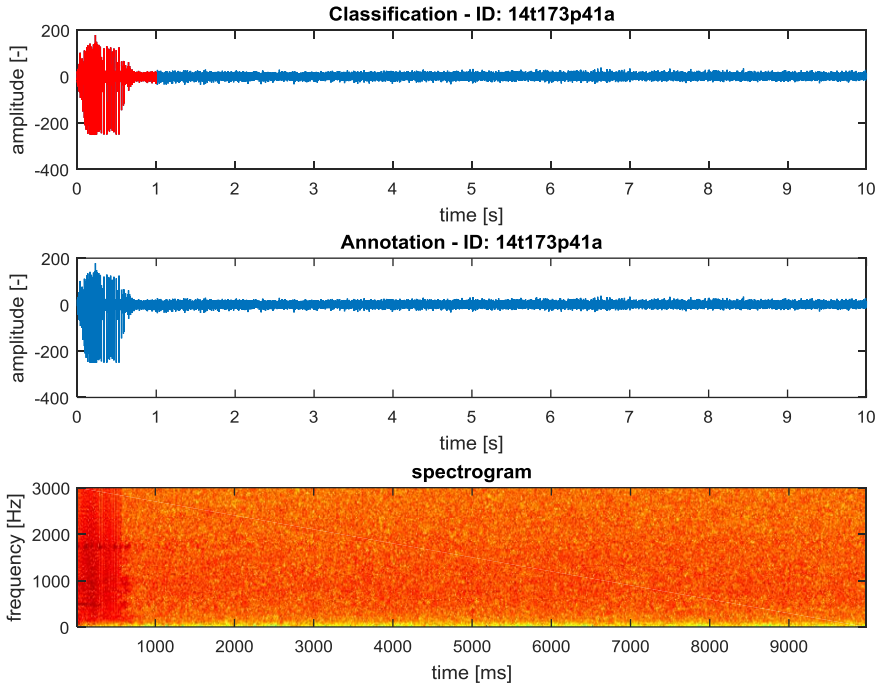


FIGURE 4.4: AN ARTEFACT DETECTED BY THE CLASSIFIER, BUT MISSED BY THE ANNOTATION

An example of an error in the annotation. The obvious artefact is unlabelled.

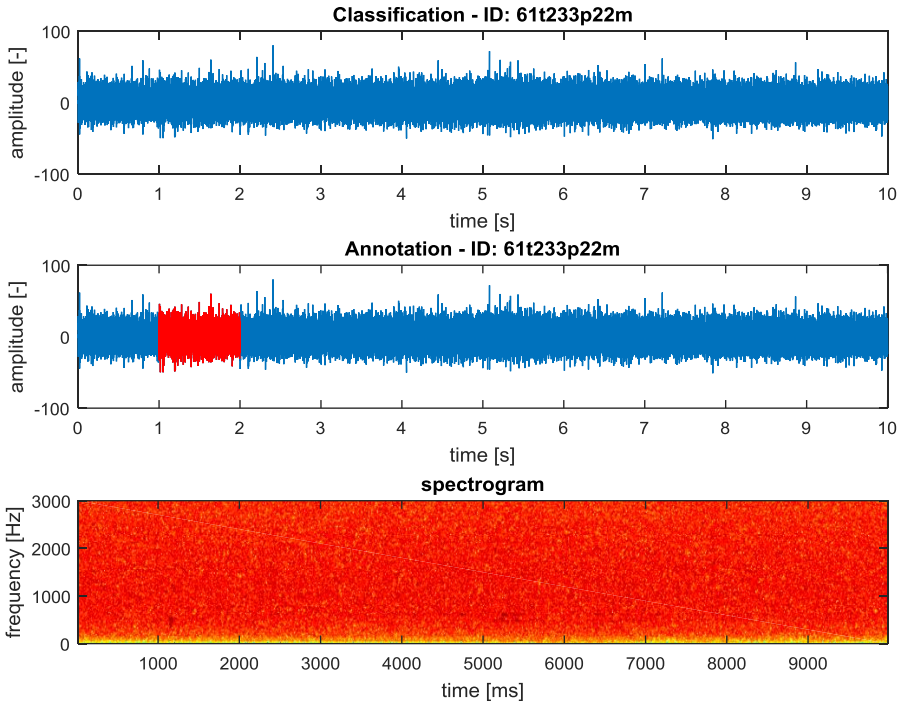


FIGURE 4.6: A WRONG ANNOTATION OF THE ARTEFACT

An example of an error in the annotation. This segment has no significant variations in amplitude and it is clearly not a power artefact.

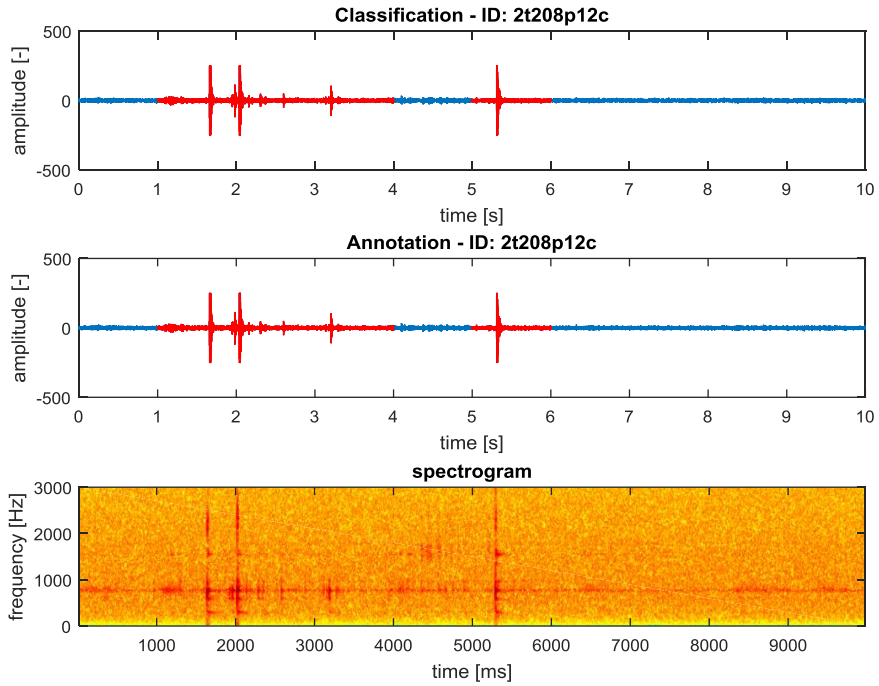


FIGURE 4.5: THE TECHNICAL ARTEFACT IS NOT DETECTED

A perfect detection of a power artefacts in the signal. The technical artefact that lasted for the whole recording and was not detected.

5 Machine-learning method

Machine learning is a comprehensive subfield of computer science that deals with algorithms that are able to learn from data and then make predictions on new samples. In the supervised paradigm, the algorithm is building a model – classifier – from data given as a training set. Machine learning is divided into two main groups. The first one is called supervised learning. The training set is provided with labels or costs of each sample. The information about the cost or the label is given usually by manual labelling of the training data. The second group is called unsupervised learning. It uses no prior knowledge about the data in the training set. The algorithm is trying to build a model using clustering – grouping data samples with similar properties. Supervised learning is suitable for example for weather forecasting. The model can be based on previous observations of weather in the past, for example the atmospheric pressure, humidity, wind direction and strength, the season, the weather in previous years and other. Unsupervised learning is suitable for data mining, thus searching for patterns and structures in the data.

The design cycle of the classifier with supervised learning includes data collection and annotation, feature extraction and selection, model selection, training, testing and evaluation [10].

FIGURE 5.1: THE DESIGN CYCLE OF THE CLASSIFIER WITH SUPERVISED LEARNING

5.1 Basic idea for this method

All methods mentioned above used large segments (from 0.5 to 1 second). My intention was to create a classifier that could be more precise than that. As seen before, not every artefact is easy to spot in the time-domain of the signal but most are clearly visible in the spectrogram. Therefore, my classifier is based on features extracted from the spectrogram. This means that the signal is divided into segments – there is no other way to extract the features from the signal, only if the signal itself was considered to be a feature. We selected a fast Fourier transform (FFT) calculated from 1000 samples windows with 50% overlap from which we got 479 segments for a 10 second signal sampled at 24 kHz. Every segment is approximately 21 ms long which is fine enough for our intentions.

Every feature mentioned further in this chapter is defined to start with the vector $\boldsymbol{s}$,

$$\boldsymbol{s} = \{\boldsymbol{s_1}, \boldsymbol{s_2}, \dots, \boldsymbol{s_N}\}, \tag{22}$$

evaluated from vector $\boldsymbol{x}$,

$$\boldsymbol{x} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}, \tag{23}$$

of N 50% overlapping segments created from the input signal. The elements of the vector $\boldsymbol{s}$ are evaluated as

$$\boldsymbol{s_i} = |\boldsymbol{FFT}(\boldsymbol{x}_i)|^2, i \in \langle 1, N \rangle, \tag{24}$$

where $\boldsymbol{FFT}(\boldsymbol{x}_i)$ is the fast Fourier transform of signal $\boldsymbol{x}_i$.

We can also define the vector $\boldsymbol{a}$ as a vector of annotations related to segments from the vector $\boldsymbol{x}$, where 1 at position with the index i means that the segment $\boldsymbol{x}_i$ contains an artefact and 0 at position with the index i means that the segment $\boldsymbol{x}_i$ does not contain an artefact.

5.2 Data collection and annotation

The data was selected from the database of 18384 MER signals described in Section 4.1 and the test and the training set were obtained the same way as in Section 4.2 – keeping the distribution of the nuclei where the signals were recorded similar to the distribution of the whole database. The only difference from Chapter 4 is that a larger database of 350 training and 150 test signals was selected for this method. Also this method uses a much finer signal annotation than the 10 segments used previously. Therefore a new, more accurate annotation had to be made.

5.2.1 Annotation

The annotation was performed using a MATLAB function written by my supervisor Ing. Eduard Bakštein. This function provided a visualization of the time-domain and the spectrogram of the signal with fixed x axis, thus zooming affected both graphs. The artefacts were labelled simply by typing the number of the artefact types and then clicking into the time-

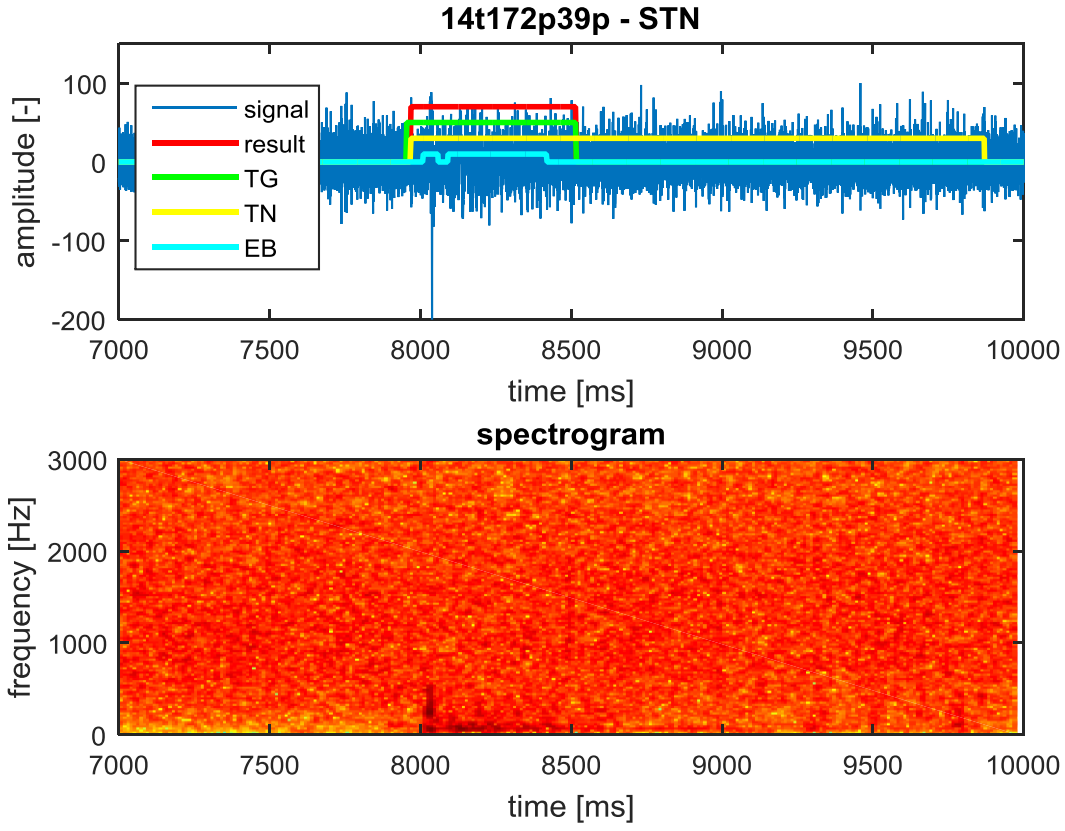


FIGURE 5.2: THE COMPARISON OF THE ANNOTATION

Differences in the annotation between three evaluators. Because it is hard to tell whether an artefact is present or not, it is impossible for several different evaluators to come to the same conclusion. In situations like this, majority vote helps to improve the reliability of the annotation.

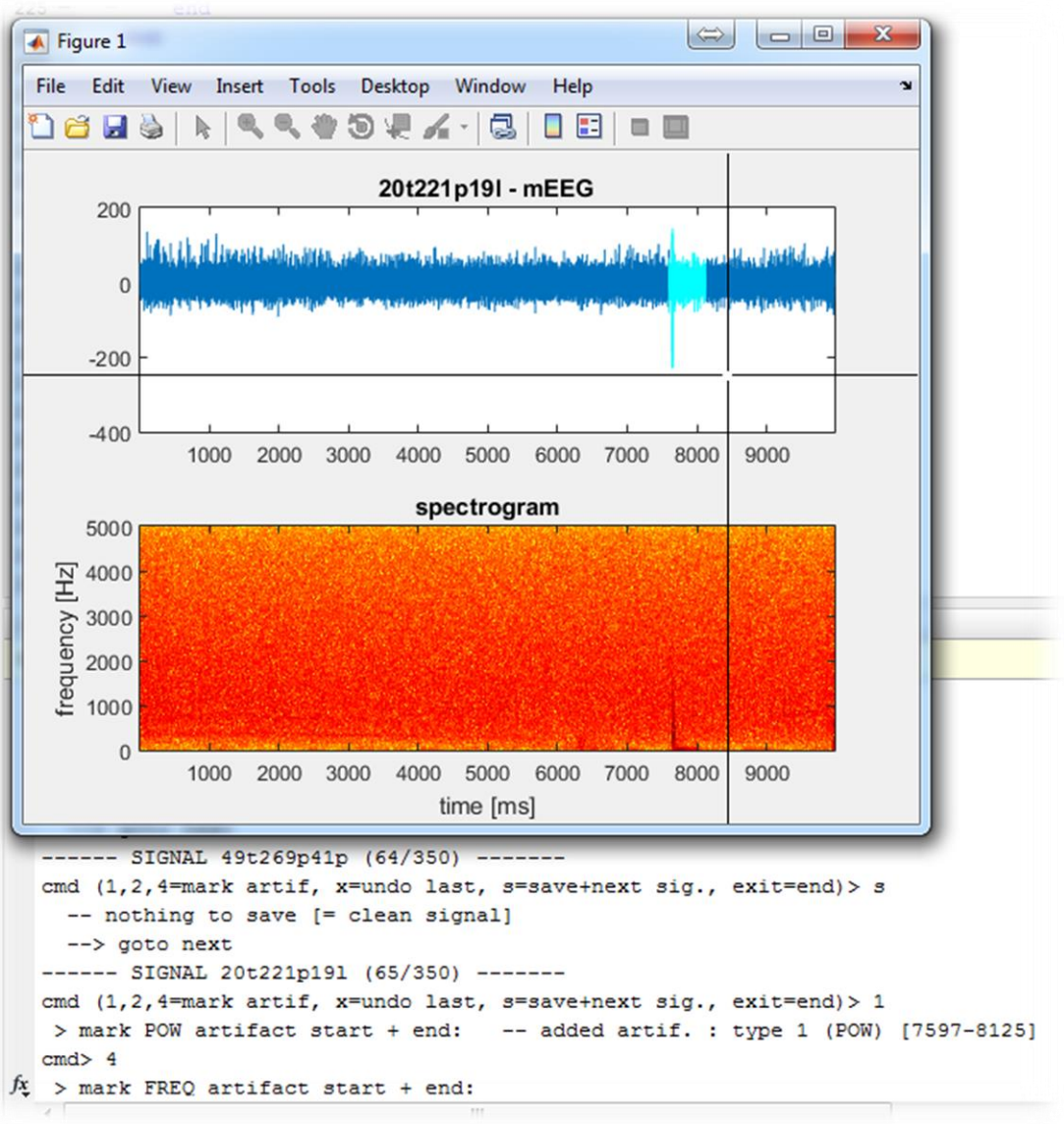


FIGURE 5.3: THE ANNOTATION TOOL

The cyan part of the graph represents the annotated power artefact. The black cross is used to define the start and the end of the artefact. The tool is operated in the command line.

domain of the signal. Three evaluators – I, my supervisor and my colleague Tomáš Grubhoffer – labelled the same data. Figure 5.3 shows the tool during the annotation.

I have ignored the types of the artefacts to preserve this problem as a binary classification and used a majority vote: two or more evaluators had to agree on the matter of labelling a part of the signal as an artefact to create the final training and test set.

The results of the annotation were sufficient. Both in the training set and in the test set all 3 evaluators agreed on only about 40% of the artefacts and the other 60% were usually agreed on by me and one of the other evaluators. This might have been caused by the difficulty of the definition of the artefact's borders. However, with the final annotation the total agreement (clean and artefacts) of the evaluators was 86% for me and 96% for the other 2 evaluators. The comparison of the annotation of different evaluators can be seen in Figure 5.2.

One of the problems I encountered during the inspection of the results was labelling the constant frequency artefacts that lasted for the whole recording. Some of them were labelled and some of them were not. Because of the previously mentioned problem with the artefact's border definition, the surroundings of the artefact might also be annotated as an artefact. This introduces an error that might make the separation of the 2 classes – clean signal and artefact – impossible.

5.3 Feature extraction and selection

The complexity and difficulty of our problem inheres in the feature extraction. In the example of the weather forecast we could instantly get the information from sensors that could be used as features but in our classification we only have signals. That means that we have to find the properties of the signal or the properties of something derived from the signal - such as the spectrogram. As mentioned above in Section 5.1, all features in our classifier were evaluated from the spectrogram. The spectrogram is also cut from the top at about 4790 Hz because everything above 5 kHz is filtered by the band-pass filter mentioned in Section 3.1.

The selection of the features is crucial for the final classifier. For example, in weather forecasting we probably would not care about the actual dollar exchange rate. Bad features can introduce noises to our classifier and therefore have a negative impact on the results.

5.3.1 ROC and AUC

One of the ways of measuring the performance of features is using the Receiver Operating Characteristic (ROC) curve and the Area Under Curve (AUC). The ROC curve is calculated from the true positive and the false positive rate of different thresholds in the classifier. In the case of feature selection the threshold is simply shifted between the minimum and the maximum value of the feature. The area under the ROC curve (AUC) is also a good representation of the feature performance. If the AUC is less than 0.5, the feature is worse than chance and if the AUC is 0.5 it is exactly the same as chance. Generally the AUC should be at least 0.7 or more to be able to say that the feature is good.

5.3.2 Feature 1: Standard deviation of frequency spectrum

The first feature is very similar to the feature used in Section 4.3 but instead of the standard deviation (STD) of amplitude, the STD of frequency spectrum is evaluated. The idea of this feature is based on an observation that showed that the frequency spectrum of contaminated parts of the signal contains high values at contaminated frequencies, thus the STD is higher.

The vector of features $\boldsymbol{r}$ is evaluated as the STD of every element of the vector $\boldsymbol{s}$ defined in Section 5.1 as

$$r_i = STD(\boldsymbol{s_i}), i \in \langle 1, N \rangle, \tag{25}$$

$$\boldsymbol{r} = \{r_1, r_2, \dots, r_N\}. \tag{26}$$

Figure 5.4 shows feature values in relation with the spectrogram. The values are significantly higher during the time period of the artefact.

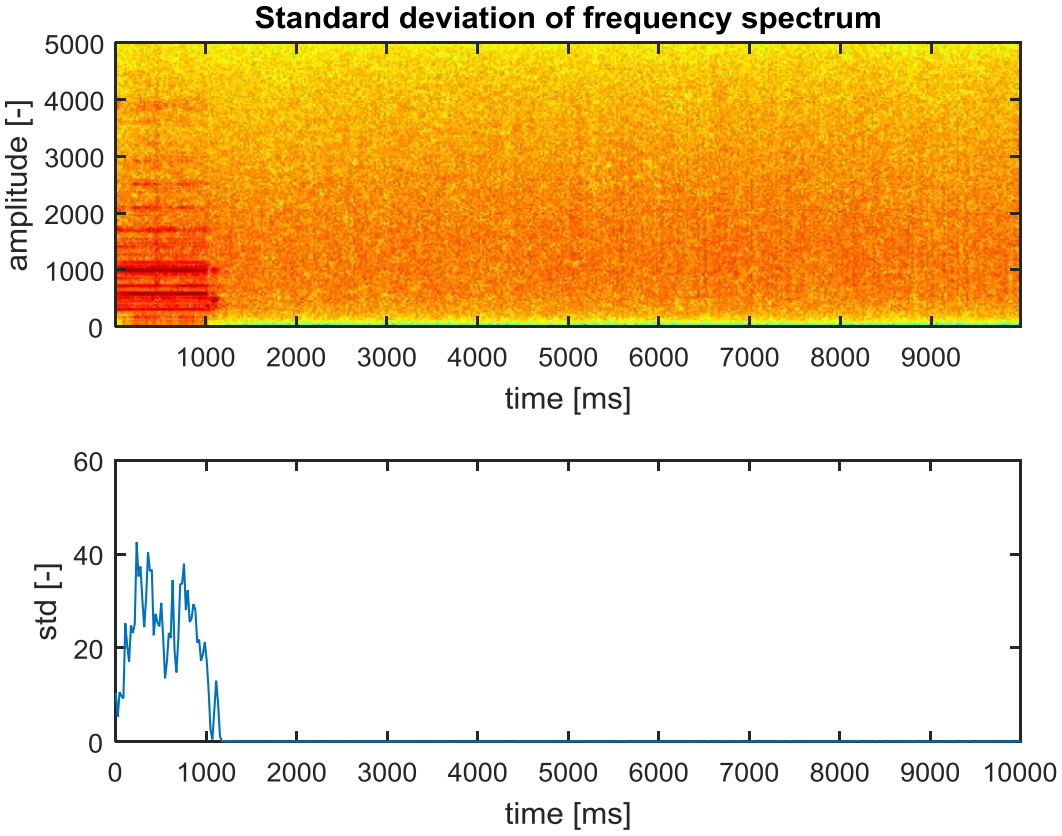


FIGURE 5.4: FEATURE 1 CALCULATED FROM A SIGNAL WITH A GREAT ARTEFACT

This figure shows the behaviour of the feature. It is obvious that its value is significantly higher during the artefact.

5.3.3 Feature 2: Kolmogorov-Smirnov difference from mean frequency spectrum

During the evaluation of this feature, the mean power spectrum $\boldsymbol{mFS}$ first has to be calculated from the training data. The segments of the signal labelled as clean are selected and the normalized power spectral density is computed for each segment according to Equation (24). $\boldsymbol{mFS}$ is the mean of all evaluated power spectrums. The spectrum of the tested segment is then compared to the $\boldsymbol{mFS}$ using Kolmogorov-Smirnov test.

The two-Sample Kolmogorov-Smirnov test (KS2 test) is a non-parametric statistical test used to determine whether two datasets differ significantly. It is done by testing the maximum difference between the empirical cumulative distribution functions (CDFs) of the two datasets.

The CDF is the probability that the value of a random variable is less than the given value. The CDF function $\widehat{\boldsymbol{F}}(x, \boldsymbol{y})$ for the function (vector) $\boldsymbol{y}$ is defined [11] as

$$\widehat{\boldsymbol{F}}(x, \boldsymbol{y}) = \widehat{\boldsymbol{F}} = \frac{1}{N}\sum_{i=1}^{N} \gamma\{y_i < x\}, \tag{27}$$

where y_i is i-th element of function $\boldsymbol{y}$, N is number of elements of function $\boldsymbol{y}$ and

$$\gamma\{y_i < x\} = \begin{cases} 1 & if\ y_i < x \\ 0 & otherwise \end{cases}. \tag{28}$$

As an analogy we can see the relationship between the probability density function (PDF) and the CDF in Figure 5.5.

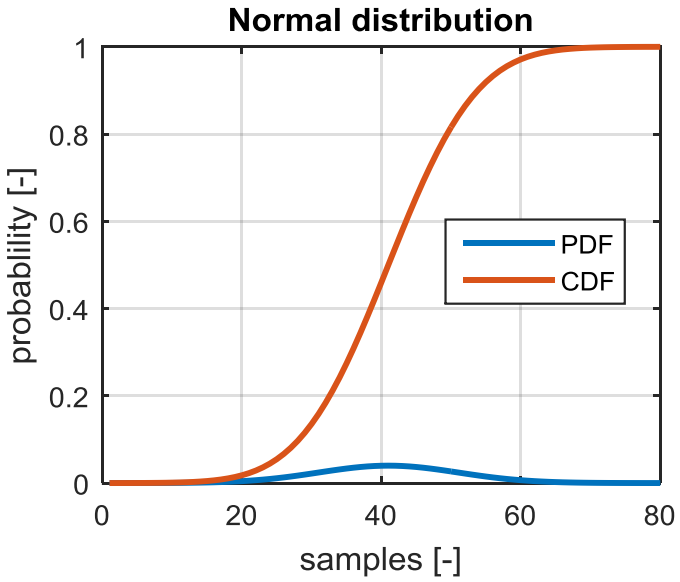


FIGURE 5.5: PDF AND CDF OF THE FUNCTION WITH NORMAL DISTRIBUTION

This figure shows that CDF is an integral of PDF

After evaluating the CDF of both functions, the KS2 test finds the maximal difference between those functions $D(\hat{F}_1, \hat{F}_2)$ as

$$D(\hat{F}_1, \hat{F}_2) = \max_x\left(\left|\hat{F}_1(x, \boldsymbol{y}) - \hat{F}_2(x, \boldsymbol{y})\right|\right). \tag{29}$$

The KS2 test now uses the table to determine the statistical significance of this value in order to reject or pass the null hypothesis. However, this feature uses only the maximal difference D.

The vector of features $\boldsymbol{r}$ is evaluated using the vectors $\boldsymbol{s}$ and $\boldsymbol{a}$ defined in Section 5.1. Before any feature evaluation can start, the $\boldsymbol{mFS}$ is evaluated from the training set as described above.

Firstly, every element of $\boldsymbol{s}$ is normalised by being divided by its sum and stored back in the vector $\boldsymbol{s}$. Then every element of the vector $\boldsymbol{s}$ is compared with the $\boldsymbol{mFS}$ using the KS2 test taking the difference $D(\hat{F}_1, \hat{F}_2)$ as the result and storing it into the vector of features $\boldsymbol{r}$ as

$$r_i = D(\hat{F}_1(x, \boldsymbol{s_i}), \hat{F}_2(x, \boldsymbol{mFS})), i \in \langle 1, N\rangle, \tag{30}$$

$$\boldsymbol{r} = \{r_1, r_2, \dots, r_N\}. \tag{31}$$

Figure 5.6 shows feature values in relation with the spectrogram. The values are significantly higher during the time period of the artefact.

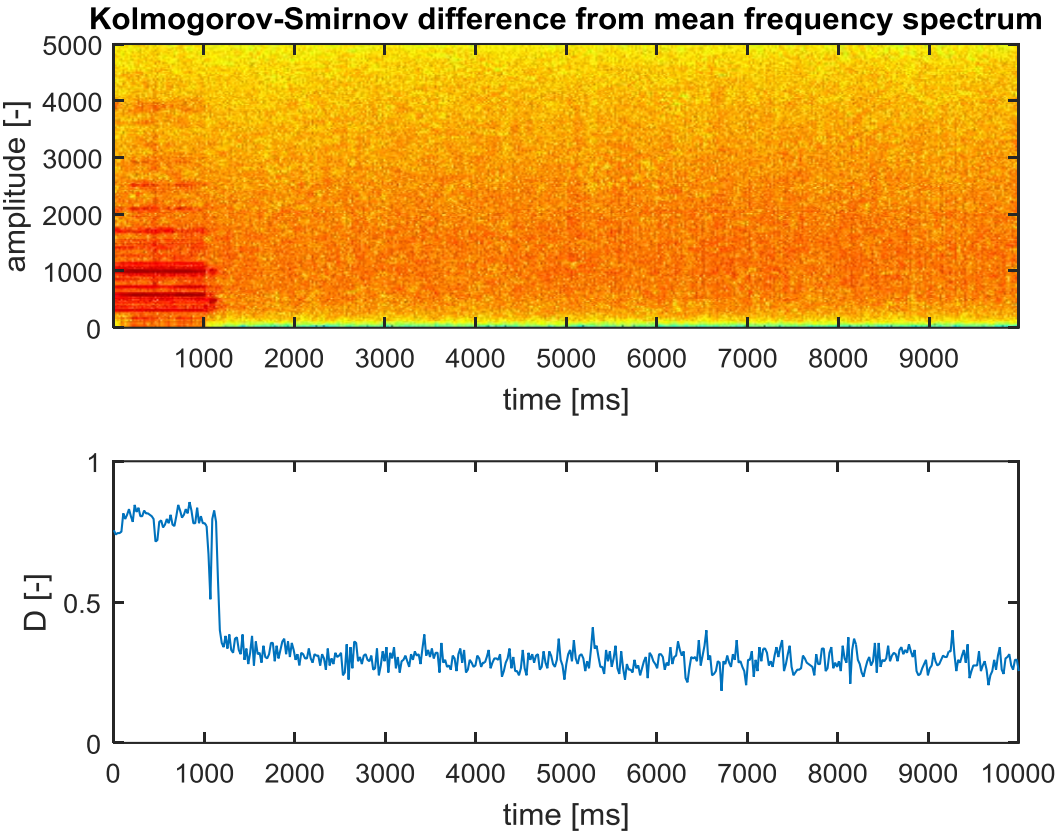


FIGURE 5.6: FEATURE 2 CALCULATED FROM A SIGNAL WITH A GREAT ARTEFACT

This figure shows the behaviour of the feature. It is obvious that its value is higher during the artefact.

5.3.4 Feature 3: Maximal difference from the mean frequency spectrum

This feature is based on the presumption that the frequency spectrum of segments with a clean signal have a similar shape and the frequency spectrums of segments with artefacts greatly differs. This feature uses the mean of the frequency spectrums of clean signals $\boldsymbol{mFS}$ evaluated in Section 5.3.3.

The vector of features $\boldsymbol{r}$ is evaluated from the vector $\boldsymbol{s}$ defined in Section 5.1 as

$$r_i = \max\left(\left|\frac{\boldsymbol{s_i}}{\sum \boldsymbol{s_i}} - \boldsymbol{mFS}\right|\right), i \in \langle 1, N\rangle, \tag{32}$$

$$\boldsymbol{r} = \{r_1, r_2, \dots, r_N\}. \tag{33}$$

Figure 5.7 shows feature values in relation with the spectrogram. The values are significantly higher during the time period of the artefact.

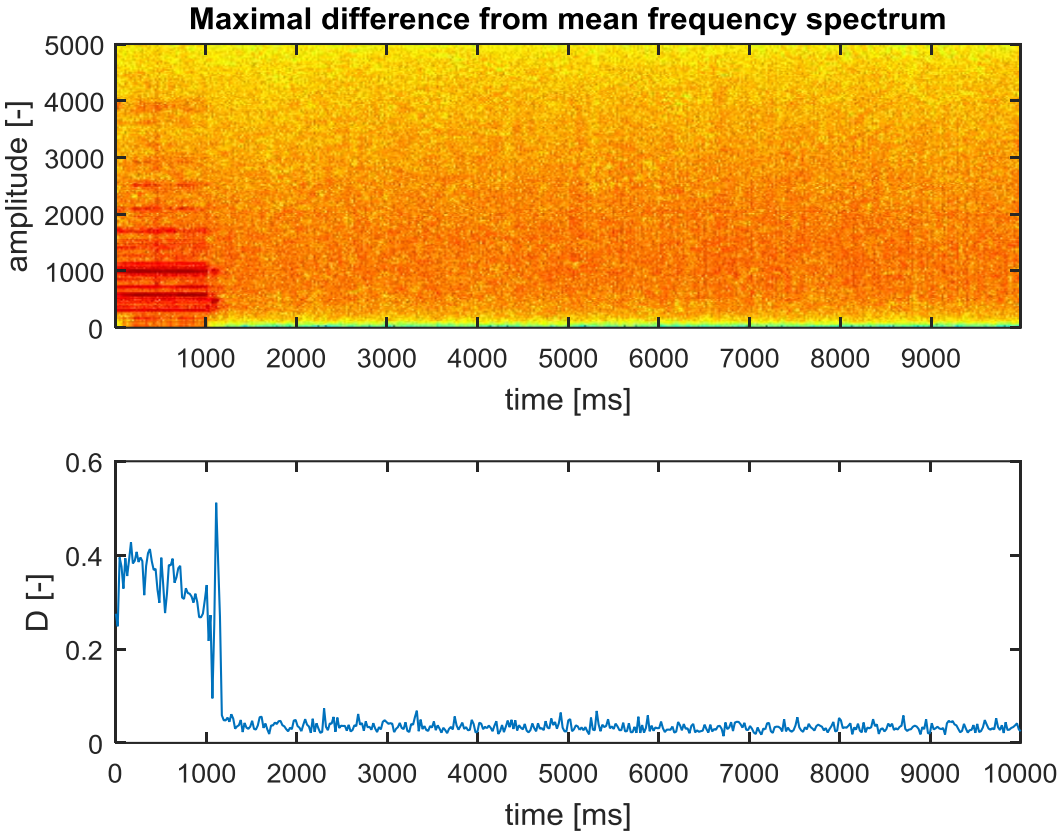


FIGURE 5.7: FEATURE 3 CALCULATED FROM A SIGNAL WITH A GREAT ARTEFACT

This figure shows the behaviour of the feature. It is obvious that its value is higher during the artefact.

5.3.5 Feature selection

The performance of features of certain problems can be measured using the ROC and the AUC mentioned in Section 5.3.1. ROC curves and AUC values in Figure 5.8 are calculated from all of the 3 features evaluated from the training data. Feature 3 performed the best. All 3 features will be used during the modelling.

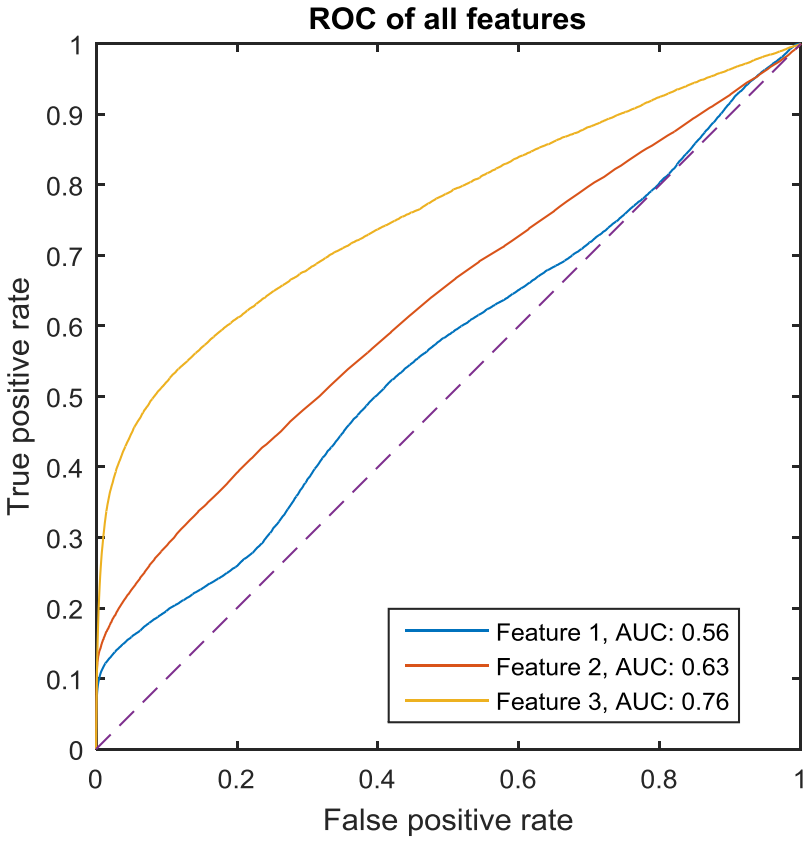


FIGURE 5.8: ROC CURVES AND THE AUC OF ALL FEATURES

5.4 Model selection

The model selection is one of the most crucial parts of the classifier designing. Every model has its advantages and disadvantages. To properly select the model we have to consider the properties of our data. Our problem is binary, thus it does not need models specialised for multiclass classification. One of the most important properties of our data is that our classes are imbalanced. According to the annotation, we only have 16.7% of samples with artefacts and 83.3% of clean samples. That means that if everything was classified as a clean signal, we would still get the accuracy of 83.3%.

5.4.1 Boosting

The boosting is a technique that improves the accuracy of any learning algorithm. It is based on creating an ensemble of weak learners [10]. For better understanding, an example of boosting with 3 component classifiers for a binary problem is presented.

We start with the training set D with n samples. Firstly, the algorithm creates a subset of n_1 random samples from the training set D and creates training set D_1. The classifier C_1 is trained from the training set D_1. The decision trees, kNN and discriminant analysis are mostly used as component classifiers. These classifiers are called weak learners because they have minimum requirements for having an accuracy higher than chance. If this learner has a high

accuracy, the problem is too simple and there is no need for boosting. A new training set D_2 is created from the remaining samples in the training set D to satisfy the condition that half of the samples in D_2 are misclassified by C_1 and half of them are classified correctly. The classifier C_2 is trained from the training set D_2. The last training set D_3 is created from the remaining samples in the training set D that have different classifications in both previously evaluated classifiers C_1 and C_2. The last classifier C_3 is trained from the training set D_3. During the classification, the new sample is first classified with the classifiers C_1 and C_2. If they return the same class, the class is taken as the result. If they disagree, the sample is classified by the classifier C_3.

One of the questions is how to choose n_1 to use as many samples from the training set D as possible. The answer varies according to the difficulty of the problem. In practice, this problem is solved by running this boosting method several times and adjusting n_1. The boosting method described above can be modified to use with any number of component classifiers and it can also be applied to multiclass problems.

5.4.2 AdaBoost

The AdaBoost is the most popular variant of boosting methods. Its name stands for "adaptive boosting", because it allows adding weak learners until some cost (for example a training error) is minimised under the defined value and every iteration weights the samples from the training set to select problematic samples for the next component classifier [10].

The algorithm starts with the training set D with N samples. The samples from the training set D have uniformly distributed weights at the beginning of the algorithm. On each iteration k, the classifier C_k is trained from the training set D_k which is randomly selected from the training set D according to the vector of the weight of samples $\boldsymbol{W_k}$ – the greater the weight

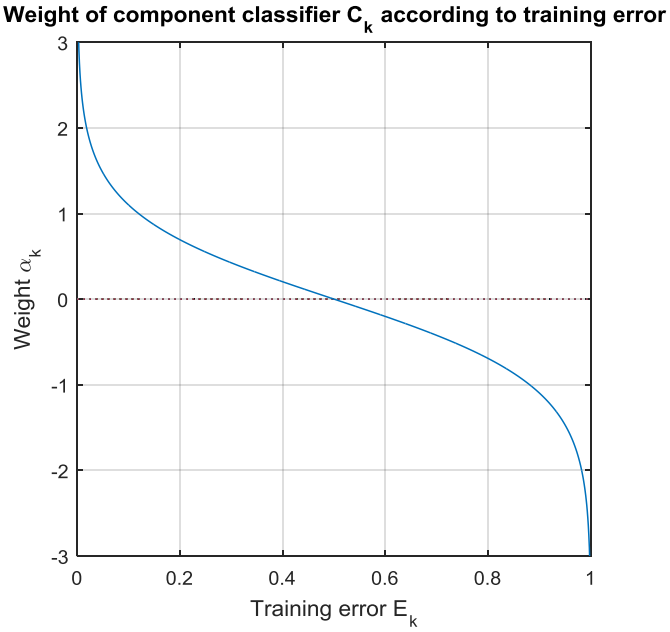


FIGURE 5.9: THE WEIGHT OF A COMPONENT CLASSIFIER BASED ON A TRAINING ERROR

It is clear from this figure that if the classifier has a training error higher than 0.5, the weight of the classifier is negative.

of i[th] sample $W_k(i)$, the higher the probability that sample will be chosen. The training error E_k of C_k is measured on the training set D. The weight of the component classifier α_k is calculated from the training error E_k as

$$\alpha_k = \frac{1}{2}\ln\left(\frac{1-E_k}{E_k}\right). \tag{34}$$

The weights of samples are updated as

$$W_{k+1}(i) = W_k(i)e^{\alpha_k},\ i \in \langle 1, N\rangle, \tag{35}$$

if i[th] sample was not correctly classified by C_k and

$$W_{k+1}(i) = W_k(i)e^{-\alpha_k},\ i \in \langle 1, N\rangle, \tag{36}$$

if i[th] sample was classified correctly. The vector of the sample weights $\boldsymbol{W_{k+1}}$ is then normalized by being divided by its own sum. This procedure is repeated until the training error E_k is smaller than the defined threshold or until k reaches the defined k_{max} value.

This way the algorithm constructs classifiers on difficult samples. The final classification of the binary problem $H(\boldsymbol{x})$ of the sample $\boldsymbol{x}$ is determined as

$$H(\boldsymbol{x}) = \text{sign}\left[\sum_{k=1}^{k_{max}} \alpha_k h_k(\boldsymbol{x})\right], \tag{37}$$

where $h_k(\boldsymbol{x})$ is the class label (1 or -1) defined given to the sample $\boldsymbol{x}$ by the classification of C_k.

The MATLAB implementation of the AdaBoost introduces another parameter – the learning rate μ. This parameter has values from the interval $(0, 1\rangle$ and represents the "speed" of learning. It simply multiplies α_k in every iteration to lower its influence. Even though there are some studies [13] that contradict this opinion, it is said that a lower learning rate should prevent overfitting. The value 0.1 (which is used often, for example it is default value in the MATLAB's Classification Learner) means that it is slowing down the learning process 10 times.

5.4.3 RUSBoost

Although the methods mentioned above are models for building very efficient classifiers, they do not solve the problem with skewed data – such as in our dataset, where the clean signal class is heavily prevalent. This problem is usually solved with so-called oversampling or undersampling.

Oversampling is a method based on generating samples from the minority class to compensate its lack. One of these methods is called SMOTE, described in [14]. It is using extrapolation to generate new samples to the minority class. Its drawback is a computational complexity, especially when used in the boosting algorithm. Also, great attention has to be paid to the extrapolation method so that the simulated samples are sufficiently representative of their class and no additional noise is introduced.

Undersampling is a method based on removing samples from the majority class. Simply removing a part of the training data set would be a great loss of information, but in conjunction with the boosting algorithm, the loss can be minimal. This is how the RUSBoost, based on

random undersampling (RUS) was created. Random undersampling randomly removes samples from the majority class to balance the classes.

The RUSBoost expands the AdaBoost so that at the beginning of every iteration k the dataset D_k is created using random undersampling, but it still respects weights of samples [15]. This way all samples have a chance to be used during the whole procedure – in one of the component classifiers. The computational complexity is also much smaller and according to [15] the overall performance on skewed databases is better.

5.4.4 Decision trees

Weak learners used by the AdaBoost and the RUSBoost are decision trees. Decision trees are classifying samples by a sequence of questions. The "tree" is an oriented or disoriented graph which does not contain a circle. A tree with n nodes has n-1 edges [16]. The "decision tree" is a disoriented tree whose edges stands for decisions (except from the edges that lead from the previous node) and nodes stands for questions except nodes that have only 1 edge. Those are called leafs – classes. Decision trees used for classification are mostly binary – every question has only 2 decisions. For example, a question in a problem with weather classification can be "Is humidity more than 64%?" and decisions are yes and no. Such decision leads to another question or to a leaf that labels the sample with a class – in this case for example if the answer is yes, the next node can be a leaf labelling sample as rain.

During the formation of the three from the training set we need to decide what feature we should ask first. Should we ask about the atmospheric pressure or humidity first? There are several measurements of impurity (a decision that separates samples with only 1 class is completely "pure"). The most popular is the entropy impurity $i(N)$ of node N evaluated as

$$i(N) = -\sum_j P(\omega_j) \log_2 P(\omega_j), \tag{38}$$

where $P(\omega_j)$ is a fraction of samples in the node N from the class ω_j [10]. As we can see, if $P(\omega_j) = 1$, impurity $i(N) = 0$, thus the node is completely pure.

Several more techniques such as pruning – elimination of branches with too few samples to prevent overfitting – are used in decision tree classifications, but more details are beyond the focus of this work.

5.5 Results

I selected the decision trees, the AdaBoost and the RUSBoost and compared their results. The decision trees were selected because they are used in both boosting algorithms as weak learners, thus we can see the impact of boosting on our problem. The RUSBoost is selected so that we can see the impact of improvements for the training using a skewed training set against the AdaBoost. The decision trees are trained with default settings in the MATLAB, the AdaBoost and the RUSBoost are trained twice with a different learning rate LR – 0.1 and 1. All bosting algorithms use 200 weak learners. The results can be seen in Table 5.1. The AUC of these results is calculated from the ROC curves in Figure 5.10. There is no way to adjust the thresholds during the classification of training data, thus the curves are only a weak approximation – the real AUC would be slightly higher. However, the AUC from those curves gives similar result as the F1 score.

Model	Parameters	True positive	False positive	False negative	True negative	Accuracy	AUC	F1 score
Decision Trees	None	3799	7843	3301	54814	84.02%	0.635	0.405
AdaBoost	LR = 0.1	3198	8444	**361**	**57754**	**87.38%**	0.634	0.421
AdaBoost	LR = 1	3591	8051	**556**	**57559**	**87.66%**	0.649	0.455
RUSBoost	LR = 0.1	**4893**	**6749**	2374	55741	86,92%	**0.690**	**0.518**
RUSBoost	LR = 1	**4850**	**6792**	2252	55863	87.03%	**0.689**	**0.517**

TABLE 5.1: THE RESULTS OF THE CLASSIFICATION OF THE TEST SET

The best results of every column are bold and underlined, the second best results are only bold. RUSBoost is giving the best results for the positive class (artefact) detection at the price of more false negative (artefacts labelled as clean signals) detections.

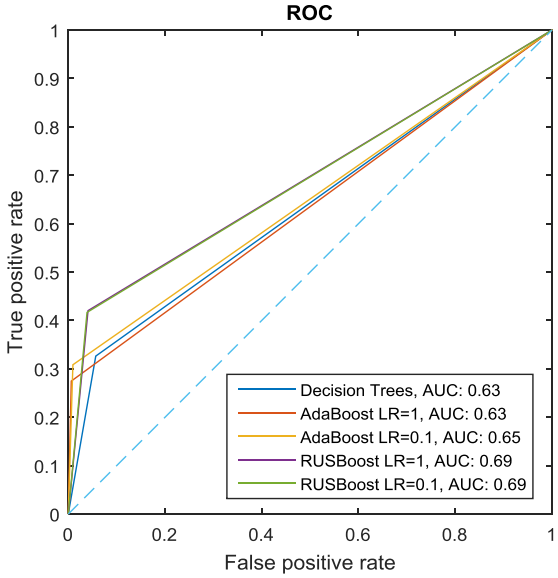


FIGURE 5.10: ROC OF THE TRAINING SET

The approximation of ROC for all classifiers

We can see that as the robustness and specialization of algorithm grows, the results are getting better. The influence of the learning rate parameter is questionable. It has no significant effect on the RUSBoost results and it actually deteriorates the results of the AdaBoost.

The results are not very satisfying, mainly because of the higher occurrence of false positive detections than the true positive ones. There are several possibilities that might contribute to this failure. The annotation has some problems as mentioned in Section 5.2.1, especially there is a problem with artefact borders that cause great problems during learning. After some investigation, I found out that the biggest problem is the length of the segments. Their length is about 20 ms, which is too small for a realistic prediction. This problem manifests itself as many small peaks in a great amount of signals, especially the signals recorded in STN. See Figure 5.11. However, we can simply take the current annotation and the classification, look at every second of the signal and every second that contains more than 12% of artefacts label as artefacts. The percentage at which the second is considered to be an artefact was found by gradual testing of values from 1% to 30% on classifications of the RUSBoost. The percentage with the best F1 score was selected.

As we can see from the new results in Table 5.2, they are much more positive. That is mainly because most of those undesired peaks were filtered during the detection as can be seen in Figure 5.11. The true positive classifications occur much more often and the F1 score rises significantly. In comparison to the sensitivity and the precision of classifiers in Table 5.3, we can see that the AdaBoost has the best sensitivity, but the RUSBoost has both precision and

Model	Parameters	True positive	False positive	False negative	True negative	Accuracy	AUC	F1 score
Decision Trees	None	10830	4263	5123	49541	86.54%	0.812	0.698
AdaBoost	LR = 0.1	7767	7326	**334**	**54330**	**89.02%**	0.754	0.670
AdaBoost	LR = 1	8726	6367	**573**	**54091**	**90.05%**	0.784	0.715
RUSBoost	LR = 0.1	**11072**	**4021**	3924	50740	88.61%	**0.831**	**0.736**
RUSBoost	LR = 1	**11072**	**4021**	3924	50740	88.61%	**0.831**	**0.736**
Stat. Segments	TH = 1.3	6673	5631	9631	51046	79.09%	0.692	0.467

TABLE 5.2: THE RESULTS OF THE CLASSIFICATION OF THE TEST SET USING ONE-SECOND INTERVAL

The best results of each column are bold and underlined, the second best results are bold. The order of the results stayed intact but both RUSBoost models are now the same regardless the learning rate.
LR – Learning rate, TH – Ad-hoc threshold

Model	Precision	Sensitivity
Decision Trees	0.718	0.679
AdaBoost	0.515	0.959
AdaBoost	0.578	0.938
RUSBoost	0.734	0.738
RUSBoost	0.734	0.738

TABLE 5.3: PRECISION AND SENSITIVITY OF THE RESULTS

Even though the sensitivity of RUSBoost is lower, it is well balanced with the precision.

sensitivity well balanced. This way the detection might be useful in practice even though some improvements should still be made.

The results of the stationary segmentation method described in Section 3.5.3 are also included in Table 5.3. The segments were set to one second and the ad-hoc threshold was set to 1.3 after the optimization on the training data. We can see that this method fails in our annotation. The main reason for this might be that it is more suitable for power artefact detection and it ignores most of the frequency artefacts that do not affect the amplitude.

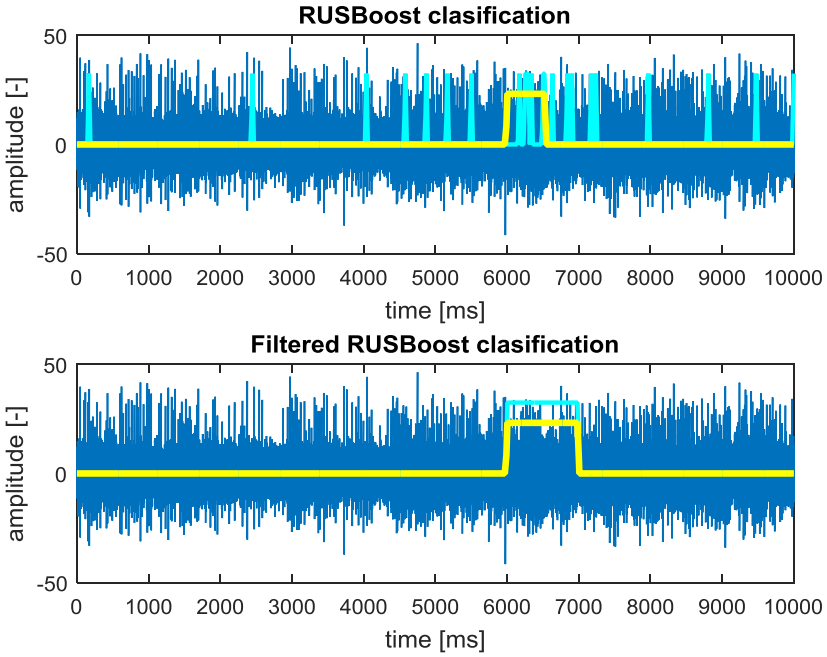


FIGURE 5.11: RESULTS OF THE CLASSIFICATION

The results of the detection using RUSBoost with LR = 0.1.The detection is blue and the annotation is yellow. The peaks are most frequent at the 6th second where the artefact is present but they are creating a lot of false detections in the rest of the signal. The filtered classification shows how the results were improved.

6 Conclusion

The main goal of this thesis was to design new methods for MER signal segmentation and artefact detection that could help researchers working with MER signals. Current methods are inadequate for large databases.

To gain basic knowledge of the issue, I have studied Parkinson's disease and its treatment, the deep brain surgery and the recording and properties of MER data. After this I have designed a method focused on the detection of power artefacts. This method proved to be too simple and inaccurate at first, however, a visual inspection of the results showed that it was sufficiently accurate and might be useful for further studies.

The second method I have designed followed common procedures of classifier designing and focused on a spectrogram of signals. At first, the annotation of artefacts was performed in cooperation with my colleagues on a database of MER signals. Then I have developed three features. Unfortunately not all of them were capable of separating the data according to the annotation precisely enough. After the feature extraction, I have selected models according to properties of training data. The RUSBoost, a boosting algorithm based on the AdaBoost and specialized for skewed data, performed the best. However, the classification showed errors that, as I believe, resulted from the size of the segments. It is problematic to agree on what is and what is not an artefact, it depends on the further usage of the data. Some applications might not be influenced by technical artefacts. Another and a much more difficult problem is determining the exact borders of artefacts.

The classification was then filtered to fit into one-second intervals and the results were very satisfying. I also proved that this method surpassed the performance of current automatic methods. The final classifier using the RUSBoost algorithm reached the accuracy of 88.61% and the F1 score of 0.736, which is a significant shift compared to the stationary segments method with the accuracy of 79.09% and the F1 score of 0.467. The suggested method shows good results and is usable for data preprocessing.

To improve this method further, I suggest to perform a more rigorous annotation and to add a higher number of stronger features that would capture other aspects of the MER artefacts and could potentially improve the classification results.

7 References

[1] A. Samii, J. G. Nutt, B. R. Ransom, “Parkinson's disease,” Lancet, vol. 363, no. 9423, pp. 1783–1793, May 2004.

[2] L. M. de Lau, M. M. Breteler, “Epidemiology of Parkinson's disease,” Lancet Neurol., vol. 5, no. 6, pp. 525–535, June 2006.

[3] K. Dvořák. *Degenerativní onemocnění mozku a míchy* [Online]. Available: https://atlases.muni.cz/atlases/stud/atl_cz/main+cnspatol+degenecns.html

[4] Z. Israel and K. J. Burchiel, *Microelectrode Recordings in Movement Disorder Surgery*, New York: Thieme, 2004.

[5] Brown University. (2008). *Parkinson's disease* [Online]. Available: http://biomed.brown.edu/Courses/BI108/BI108_2008_Groups/group07/Parkinsons.html

[6] A. Moran, I. Bar-Gad, H. Bergman, Z. Israel, “Real-time refinement of subthalamic nucleus targeting using Bayesian decision-making on the root mean square measure,” Mov Disord, vol. 21, no. 9, pp. 1425–1431, Sep. 2006.

[7] J. H. Falkenberg, J. McNames, M. Aboy, K. J. Burchiel, "Segmentation of extracellular microelectrode recordings with equal power," Annual International Conference of the IEEE EMBS - Proceedings, Cancun, Mexico, pp. 2475-2478, Sept. 2003.

[8] M. Aboy, J. H. Falkenberg, “An automatic algorithm for stationary segmentation of extracellular microelectrode recordings,” Med Biol Eng Comput, vol. 44, no. 6, pp. 511-515, May 2006.

[9] M. Navara, *Pravděpodobnost a matematická statistika*, Prague, Czech Republic: České vysoké učení technické v Praze, 2007.

[10] R. O. Duda, P. E. Hart and G. S. Stork, *Pattern Classification*, New York: Wiley, 2001.

[11] R. Castro. (2013, Feb. 2013). *Introduction and the Empirical CDF* [Online]. Available: http://www.win.tue.nl/~rmcastro/ AppStat2013/files/lecture1.pdf

[12] MarhWorks. *Two-sample Kolmogorov-Smirnov test* [Online]. Available: http://www.mathworks.com/help/stats/kstest2.html

[13] D. Mease, A. Wyner, “Evidence Contrary to the Statistical View of Boosting,” Journal of Machine Learning Research, vol. 9, pp. 131-156, 2008.

[14] N. V. Chawla, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer, “Smote: Synthetic minority oversampling technique,” Journal of Artificial Intelligence Research, vol. 16, pp. 321– 357, 2002.

[15] Seiffert, Ch., T. Khoshgoftaar, J. Hulse, and A. Napolitano, “RUSBoost: Improving clasification performance when training data is skewed,” 19th International Conference on Pattern Recognition, pp. 1–4, 2008.

[16] M. Demlová. (2010). *Grafy* [Online]. Available: http://math.feld.cvut.cz/demlova/teaching/lgr/p-lgr512.pdf

8 Appendix: CD contents

Folders:

- *TomasNovak_BachelorsThesis2015.pdf* – this thesis in the pdf format
- **scripts**: scripts and functions, require the DBS toolbox created by the Computational Neuroscience research group (http://neuro.felk.cvut.cz/)
    - **artifactAnnotationFunc**: scripts for the annotation and for logical vector corresponding to feature/signal length.
        - *segmentsDatabaseAnnotate.m* – the script for the annotation
        - *ids.mat* – IDs of signals selected for the annotation
        - *test.mat, train.mat* – the output of the annotation
        - *changeAnnotToLogicalMajorityVote*.m – change annotation to a logical vector with the length of the signal/feature and applies majority vote for the final annotation
        - *database_final.mat, database_final_spectrosize.mat* – database of IDs and the corresponding annotation
        - *makeLogVector.m*– create a logical vector, from the given annotation
    - **powerArtefactDetection**: the scripts used during the designing and testing power artefact detection.
        - *powerDatabaseSelectTN.m* – selects random signals with given nuclei distribution
        - *rawDatabase.mat* – the output of signal selection, IDs of selected signals
        - *findAnnotation.m* – adds the annotation from the database to selected signals and filters them to only contain only power artefacts
        - *adHocTest.m* – tests different values of the constant C and saves the results
        - *databaseWithAnnotation.mat, databaseWithAnnotationOnlyPow.mat* – the database of the signal IDs with the annotation
        - *databaseIdAnnotFeatResult.mat* – the database with the annotation (power and technical artefacts), evaluated feature and result of the detection
        - *databaseIdAnnotFeatResultOnlyPow.mat* – the database with the annotation (power artefacts only), evaluated the features and results of the detection
        - *detectionTestFigures.m* – plots the visualisation of the detection on all signals from the test set
        - ***detectPowerArtPow.m*** – returns a classification for the given signal, a number of intervals and an ad-hoc constant C
        - *evaluateFeaturePow.m* – evaluates features and the results of the detection for the whole database
    - **MLclassifier**: the scripts used during the classifier's designing and testing using machine learning principles
        - *createDataForClassifFromDatabase.m* - creates a vector of all samples with the annotation, usable for classifier learning
        - *database_base.mat* – a database of IDs of signals with the annotation

- *database_with_features.mat* – a database with evaluated features
- *database_withFullLen.mat* – a database of IDs of the signals with the annotation with length of the signal (for the stationary segments method)
- *databaseForClassif_FullLen.mat* – the vectors of the annotation for the test and training set, usable for classifier learning
- *evaluateFeatures.m* – evaluates features for the database of IDs with the annotation
- *feature1.m, feature2.m, feature3.m* – evaluates features for given ID
- *feature2and3mFSCalculate.m* – calculates the mFS for feature 2 and 3
- *getFeatForId.m* – evaluates all 3 features for a given signal ID
- *mFS.mat* – calculated mFS used during feature 2 and 3 evaluation
- *plotClassifForId.m* – plots classification for given signal id, classifier and features used by the classifier
- *plotCompareToAnnotation.m* – plots a comparison of detection and the annotation for all signals from the test set
- *plotSigWithFeature.m* - plots a vector of the selected feature for the selected signal
- *rougherVector.m* – filters the annotation or the detection to one-second segments
- *segmentsDatabaseSelectTN.m* – selects random signals with given nuclei distribution
- *showFeature.m* – plots a vector of the selected feature for 10 selected signals or the ROC curves for all features
- *trainAndStatRUSBoost.m* – trains the RUSBoost classifier and tests it on the test set
- *testStatSegments.m* – tests stationary segments method on the test set
- *traindAndStatTreeAdaRus.m* – trains and tests several classifiers and the saves results
- *wholeSpectro.m* – plots a spectrogram and the PSD of the signal
- ***finalClassifier.mat*** – final RUSBoost classifier, can be applied to new data as is

- **utils**
  - *dbsGetArtifData.m* – returns the annotation (integrated into database) for the given signal
  - *dbsGetDataSel.m* – returns the dataSel for given information about the signal (patient, exploration, electrode, position)
  - *dbsGetExplTrajectory4patientId* – a script converting the trajectory ID to the exploration ID
  - *otsuThreshold* – an implementation of threshold selection method from the grey-level histogram by N. Otsu