BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

REAL-TIME PROCESSING OF INTRACRANIAL EEG SIGNALS

ZPRACOVÁNÍ INTRAKRANIÁLNÍCH EEG SIGNÁLŮ V REÁLNÉM ČASE

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR
AUTOR PRÁCE

PATRIK BEGÁŇ

SUPERVISOR
VEDOUCÍ PRÁCE

Prof. Dr. Ing. JAN ČERNOCKÝ

BRNO 2021

Brno University of Technology
Faculty of Information Technology

Department of Computer Graphics and Multimedia (DCGM) Academic year 2021/2022

Bachelor's Thesis Specification

25242

Student:	**Begáň Patrik**
Programme:	Information Technology
Title:	**Real-Time Processing of Intracranial EEG Signals**
Category:	Signal Processing

Assignment:

1. Study the principles of real time processing of electrophysiological data.
2. Design an appropriate SW architecture for realtime processing of 5 kHz intracranial EEG data.
3. Implement and optimize the proposed SW architecture.
4. Implement algorithms from EPYCOM library into the architecture and assess their suitability for realtime processing.
5. Analyze algorithm performance for localization of epileptogenic tissue in online mode compared to offline mode.
6. Discuss the utility of online intracranial EEG processing.

Recommended literature:

- Cimbalnik, J., Klimes, P., Sladky, V., Nejedly, P., Jurak, P., Pail, M., Roman, R., Daniel, P., Guragain, H., Brinkmann, B., Brazdil, M., & Worrell, G. A. (2019). Multi-feature localization of epileptic foci from interictal, intracranial EEG. Clinical Neurophysiology, 130(10), 1945-1953. https://doi.org/10.1016/j.clinph.2019.07.024

Requirements for the first semester:

- Items 1 to 3, partial completion of item 4.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor:	**Černocký Jan, prof. Dr. Ing.**
Consultant:	Cimbálník Jan, Ing. Mgr., FNUSA
Head of Department:	Černocký Jan, doc. Dr. Ing.
Beginning of work:	November 1, 2021
Submission deadline:	May 11, 2022
Approval date:	November 1, 2021

Abstract

In this thesis, we designed and implemented a tool that is able to process intracranial EEG signals in real-time. That is done by applying functions for computing various iEEG biomarkers implemented in python library `Epycom` on the incoming data stream and storing the results into the database. We compared results computed by our tool against the offline computations and evaluated if real-time signal processing is suitable for clinical practice.

Abstrakt

V této práci jsme navrhli a implementovali nástroj, který je schopen zpracovávat intrakraniální EEG signály v reálném čase. To se provádí aplikací funkcí pro výpočet různých iEEG biomarkerů implementovaných v python knihovně `Epycom` na příchozí datový tok a uložením výsledků do databáze. Porovnali jsme výsledky vypočítané naším nástrojem s offline výpočty a vyhodnotili, zda je zpracování signálu v reálném čase vhodné pro klinickou praxi.

Keywords

EEG signals, epilepsy, EEG processing, intracranial EEG, real-time signal processing, relative entropy, St Anne's Hospital Brno, pre-surgical assessment, localization of epileptogenic tissue, epycom library

Klíčová slova

EEG signály, epilepsie, EEG zpracování, intrakraniální EEG, zpracování signálu v reálném čase, relativní entropie, Nemocnice u sv. Anny Brno, předoperační vyšetření, lokalizace epileptogenní tkáně, knihovna epycom

Reference

BEGÁŇ, Patrik. *Real-time processing of intracranial EEG signals*. Brno, 2021. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Prof. Dr. Ing. Jan Černocký

Rozšířený abstrakt

Tato práce vychází z potřeby vývoje nových nástrojů v oblasti farmakorezistentní léčby epilepsie. Pacienti trpící tímto typem epilepsie nereagují na běžná antiepileptika a ve většině případů je jediným způsobem, jak dosáhnout bezzáchvatového stavu, resekce epileptogenní mozkové tkáně. Přesná lokalizace epileptogenní zóny (EZ) je proto nezbytná. V rámci tohoto úkonu musí být pacientovi implantovány intrakraniální EEG elektrody po dobu až čtyř týdnů, přičemž čekání na výskyt záchvatu je dodnes jediným způsobem lokalizace EZ. V některých případech ani tento proces nestačí k označení části mozku pro resekci (tato označená část se také nazývá zóna nástupu záchvatu nebo SOZ) s dostatečnou jistotou k provedení operace. V takovém případě se předoperační posouzení ukazuje jako zbytečné. Proto je vynaloženo velké úsilí na zlepšení tohoto procesu. Klíčovými parametry jsou čas a přesnost. Tato diplomová práce, vytvořená ve spolupráci s FNUSA-ICRC, pokládá základ pro řešení obou zmíněných aspektů. Implementovali jsme nástroj v Pythonu, který dokáže zpracovat příchozí proud iEEG signálů a extrahovat z něj požadované vlastnosti iEEG v reálném čase. Vzhledem k tomu, že kombinace různých iEEG biomarkerů přináší lepší výsledky při lokalizaci SOZ [4], lze přesnost této úlohy zlepšit pomocí našeho nástroje v kombinaci se strojovým učením nebo jiným rozhodovacím algoritmem. Zpracování v reálném čase zajišťuje, že výsledky jsou k dispozici okamžitě a neurologové se mohou rozhodovat přímo během nahrávání. Na rozdíl od offline zpracování jsou při online zpracování data ukládána do databáze dobře strukturovaná pro další vyhodnocování. Náš nástroj byl testován na datech od čtyř pacientů. Tři datové sady byly dlouhé záznamy trvající několik hodin a pacienti se během doby nahrávání mohli volně pohybovat po nemocnici. Jeden byl 30 minutový záznam pacienta ležícího na lůžku. Abychom náš nástroj vyhodnotili, položili jsme čtyři otázky:

1. Zpracovává náš nástroj data stejným způsobem, jako když jsou zpracovávána offline?
2. Je nástroj dostatečně robustní pro zpracování dlouhodobých signálů?
3. Dosahuje nástroj podobného výkonu jako referenční 30 min. nahrávání v leže během uvolněném stavu?
4. Mění se schopnost lokalizovat SOZ pomocí biomarkerů v čase a může překonat referenční záznam v některých segmentech?

Vypočítané výsledky byly statisticky porovnány a vyneseny do grafů. V krátkém záznamu byly rozdíly mezi offline a online vypočítanými daty malé (otázka 1). Nástroj zpracoval všechny datové sady bez selhání nebo jiného neočekávaného chování (otázka 2). Funkce pro výpočet relativní entropie a detekci HFO byli aplikované na dlouhé datové sady proudící do našeho programu. Pro srovnání s offline výpočty bylo jako reference k dispozici pouze 30 min. offline vypočítaných dat. Výsledky byly porovnány s daty zpracovanými v reálném čase rozdělenými do 30 min dlouhých časových segmentů. Hodnoty REN po výpočtu ROC se statisticky rovnaly referenčním výsledkům z 30 min relaxovaného záznamu. Z toho vyplývá, že náš nástroj je vhodný pro analýzu dlouhodobých nahrávek s podobným výkonem jako offline analýza krátkých nahrávek (otázka 3). Hodnoty ROC HFO byly statisticky horší než referenční ROC. To může být způsobeno nižší kvalitou dlouhodobých nahrávek, které jsou vystaveny mnohem většímu šumu. Potenciální výhody našeho nástroje stále převyšují jeho nižší výkon v detekci HFO. Je to i kvůli zjištěním po analýze statistických rozdílů v jednotlivých segmentech. Navzdory celkovému výsledku byl výkon našeho

nástroje podobný offline analýze v některých saostatných segmentech při detekci HFO. To naznačuje existenci časově proměnných epileptických cyklů v mozku a nabízí možnost lepší lokalizace SOZ díky této znalosti (otázka 4).

Přestože na celém světě probíhá v oblasti EEG a epilepsie mnoho výzkumů a vývoje, zatím není na mnoha institucích plně implementováno online zpracování EEG. Náš projekt může dále poskytnout pevný základ pro další rozvoj lepších mechanismů lokalizace a predikce záchvatů v Nemocnici u sv. Anny v Brně. Navíc díky okamžitým výsledkům mohou lékaři učinit rozhodnutí pro zajištění lepšího komfortu pacienta přímo během snímání iEEG. Projekt může také pomoci s výběrem správného času pro 30 min nahrávání, kdy pacient jenom leží, což může dále usnadnit další lokalizaci. Přestože je náš nástroj ve svém současném stavu plně funkční, aby byl uveden do praxe, je třeba ještě vynaložit úsilý pro jeho vývoj.

Real-time processing of intracranial EEG signals

Declaration

I hereby declare that this Bachelor's thesis was prepared as an original work by the author under the supervision of Mr. Ján Cimbálník and Mr. Jan Černocký. The supplementary information was provided by Mr. Petr Klimeš. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis. acknowledgment

. . . .
Patrik Begáň
May 10, 2022

Acknowledgements

This project wouldn't be possible without the help of Jan Cimbálník, who became my mentor and closest collaborator. He helped me almost with every aspect of the project by providing data, tips, and consultations when things were not working. He showed me how to evaluate the project and often worked with me on the project in his free time. Big thanks deserve Mr. Černocký for guiding me throughout the whole process of writing, always giving me good tips about the text, and seeing a space for improvement. He is a busy man, yet he found time for me every time I needed him. I would also like to thank Petr Klimeš, who helped me write the theoretical part. He was guiding me in the fields of biosignals and epilepsy since they were relatively new for me before I started writing this thesis. He also covered the bureaucratic part of our cooperation on this project by ensuring a contract for me.

Contents

Glossary

Most commonly used shortcuts and abbreviations in this thesis.

AUC	area under the curve
BME	department of biomediccal engineering at St Anne's Hospital Brno
CS (detector)	Cimbálník-Stead detector; algorithm for detecting high-frequency oscillations
EEG	electroencephalogram, electroencephalograph (depending on context)
EoCG	electrocorticography
EZ	epileptogenic zone
FNUSA	Fakultní Nemocnice u Sv. Anny (St Anne's Hospital Brno)
FNUSA-ICRC	The International Clinical Research Centre
HFO	high-frequency oscillations
ictal	during epileptic seizure
iEEG	intracranial EEG
IED	interictal epileptiform discharges (also referred to as *spikes*)
interictal	in between of epileptic seizures
KL (divergence)	Kullback-Leibler divergence
MI	modulation index
MVL	mean vector length
PAC	phase-amplitude coupling
REN	relative entropy
ROC	receiver operating characteristic (curve)
sEEG	stereotactic EEG
SOS	second-order series (in relation to filtering)
SOZ	seizure onset zone
SVM	support vector machine

Chapter 1

Introduction

1.1 Motivation

Epilepsy is a disease that affects about 1 % people worldwide. Many of those people can be treated pharmacologically, yet about 30 % of epilepsy patients do not respond to any kind of conventional epileptic drug [30]. In such a case, the best option for a patient is to undergo surgical treatment, which focuses on the epileptogenic zone (EZ) resection [31]. However, only about 60% of patients remain seizure-free after administering it [9]. Pre-surgical assessment of EZ plays a key role in the outcome of surgery and much effort is being put into developing more accurate and automatized methods for localization of these zones by analyzing electroencephalographic (EEG) signals of the brain. Several EEG biomarkers can be used for the diagnosis of epilepsy, some of which are visually observable on EEG recordings, yet many others require further processing and analysis. The period of presurgical analysis, together with the EZ localization, is on average two weeks long [4] where most of the time is spent by waiting for patient's seizure. During this time, the patient's skull is penetrated with electrodes, and the patient is exposed to the risk of infection or further deterioration. Besides that, a week spent in the intensive care unit costs about 150 thousand Czech crowns.

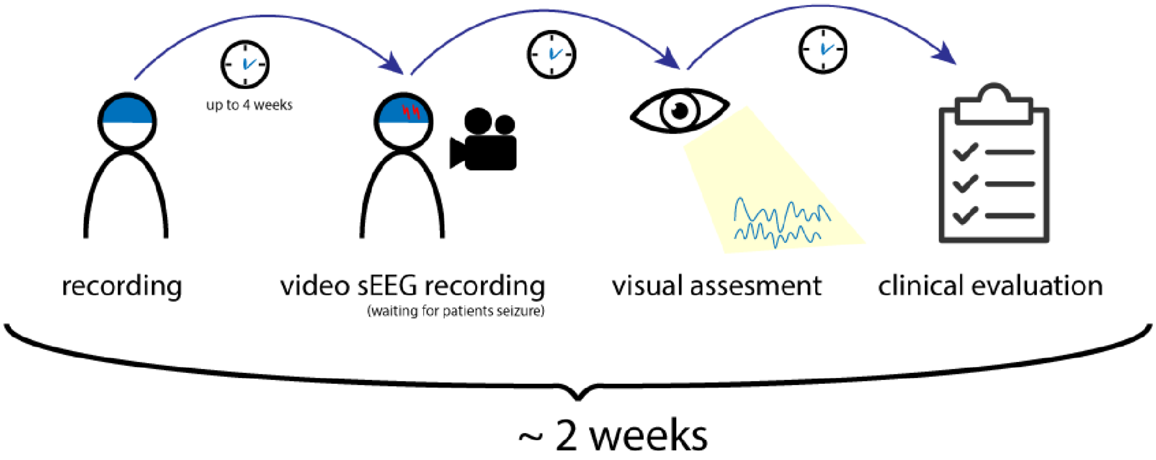


Figure 1.1: Golden standard of pre-surgical analysis.

The motivation for this thesis lies in the need for developing algorithmic tools for efficient analysis of non-observable properties of EEG signals and offering real-time data that can be used for EZ diagnosis and speeding up the whole pre-operation process. Developing a tool that could analyze intracranial signals coming in real-time, right from the inside of the head of a patient, could potentially speed up the whole process and by combining various bio-markers also improve the precision of the EZ localization, thus enhancing the overall probability for the seizure-free outcome of the surgery. Although the current golden standard – visual assessment of EEG recording, captured during the seizure (figure 1.1), is most likely not going to change during the following years, with more research on biomarkers computed from the recordings between the seizures and development of tools like the one described in this thesis, we might reach a point in the future where seizure recording will not be needed for an accurate pre-surgical evaluation and the whole process would be done under 24 hours, or even in the operating room. The thesis is based on ongoing research at The International Clinical Research Centre of St. Anne‘s University Hospital Brno (FNUSA ICRC)[1] in cooperation with the Institute of Scientific Instruments of The Czech Academy of Sciences (ISI-CAS)[2] focused on the analysis of high-frequency intracerebral EEG signals.

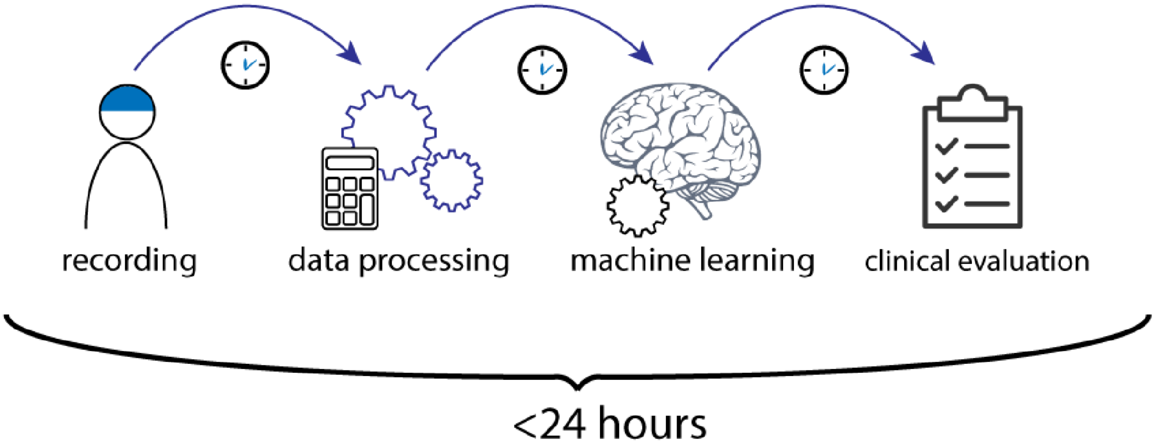


Figure 1.2: Future prospects on the process of presurgical analysis, that could be far less costly and more efficient than current golden standard.

1.2 Collaboration

This project is part of a larger cooperation between several institutions around the globe. I have been cooperating with the Computational Neuroscience research group at ISI-CAS together with The Biomedical Engineering research team (BME) from FNUSA ICRC in Brno[3] since autumn 2020 by making contributions to the python library `Epycom`[4], which contains functions for calculation of various EEG biomarkers and is also used in this project. Later, I signed a contract with ISI-CAS and started working on the already initiated, but at the time stagnating project and library `Mepior`. The knowledge base for the implemen-

[1] https://www.fnusa.cz/en/hp/

[2] http://www.isibrno.cz/en

[3] https://www.fnusa-icrc.org/en/research/research-teams/clinical-research/biomedical-engineering/

[4] https://gitlab.com/icrc-bme/epycom

tation comes from several studies published by the institutions mentioned above in further cooperation with the Mayo Clinic Department of Neurology[5], such as [4]. Therefore this work represents results coming from the cooperation of several departments across multiple institutions, rather than as a single person solution.

1.2.1 Co-workers

During the writing and implementation, I could always rely on the advice of my external supervisor Ing. Ján Cimbálník, Ph.D. [6] who has many years of experience in the research of electrophysiological signals and has worked with EEG data from epileptic patients directly. Besides, he abounds with deep technical knowledge of programming signal processing scripts and libraries. He originally initiated the whole project and introduced me to it. At the same time, he provided me with support programs and data from his lab. The second important person participating in this project is Ing. Petr Klimeš, Ph.D. [7], with many publications and rich experience in the field of EEG signals. Besides deepening my understanding of the topic and offering consulting about brain electrophysiology and epilepsy, he took care of the formal part of our collaboration, giving me a contract.

1.3 Work progress

Because of the project complexity, work was progressing through the several stages. First, I had to get familiar with project in the state as it was. That was the phase of studying all the tools, doing configurations and solving dependency problems as well as learning about the already existing architecture. Next, I started implementing my adjustments in order to make the pipeline work as expected. Together with Ján Cimbálník and Petr Klimeš, we were having regular meetings every two weeks in order to discuss the project progress, emerging questions and possible improvements. In the following phase, after the functionality of the pipeline on a single core of single machine has been tested, I implemented multi-process pipeline with the use of the tools for managing distributed nodes, described in Chapter 4. Later, I ran this enhanced pipeline on multiple CPUs at St. Anne's Hospital and performed the tool evaluation which was the last phase of the development. The distribution of work is displayed in figure 1.3.

1.4 Claims

We implemented tool processing EEG signals in real time which is processing the signals comparably well with the offline processing. Our tool is suitable for clinical use and will be put into practical use at St. Anne's Hospital Brno.

[5] https://www.mayoclinic.org/

[6] https://scholar.google.com/citations?hl=en&user=jrlYK1EAAAAJ

[7] https://scholar.google.com/citations?user=x7Iu8KoAAAAJ&hl=en&oi=sra

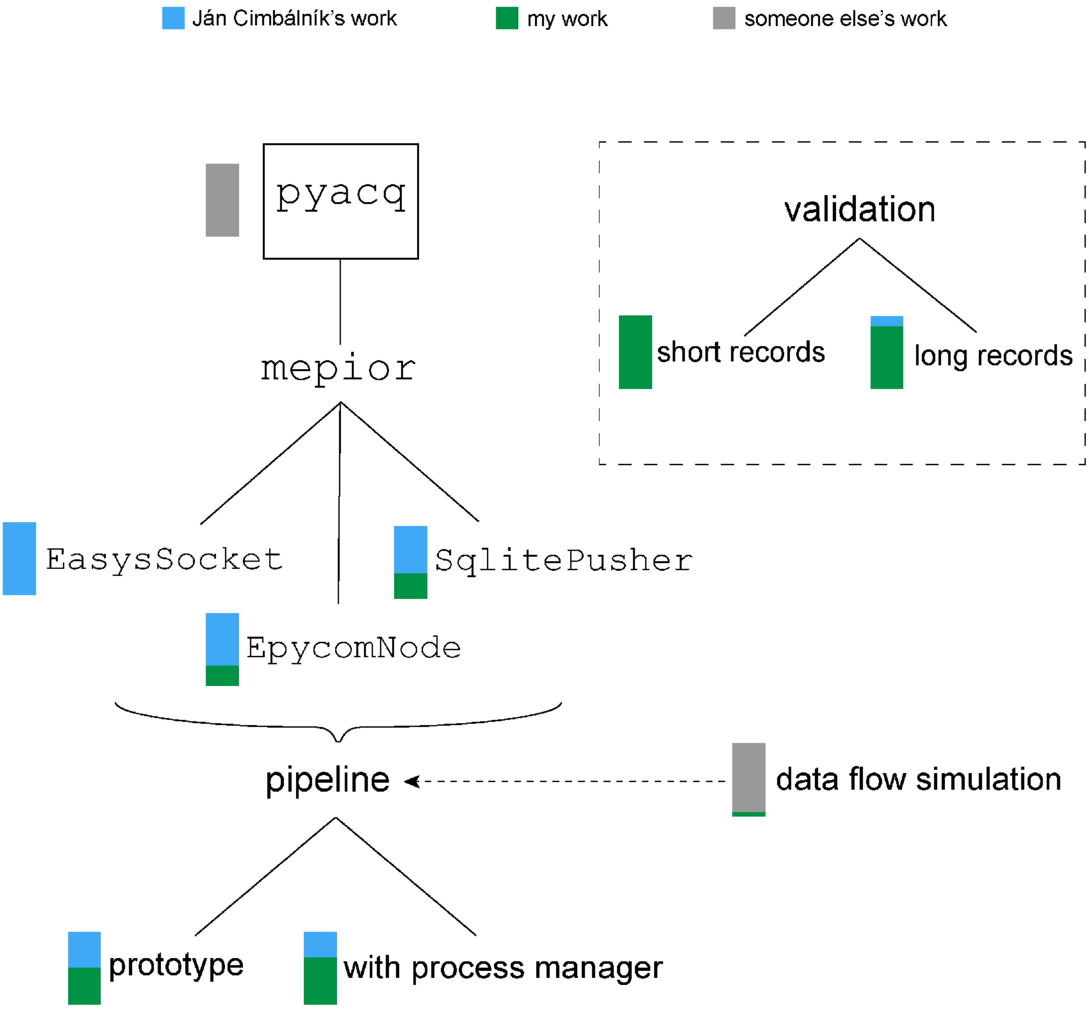


Figure 1.3: Approximate contribution to the project. On the top, `pyacq` is the library which was initially forked from GitHub and by building up on it, `mepior` library was created. The classes from mepior used in our pipeline implementation are displayed in the figure. Note, that there are other classes in mepior implemented by Ján Cimbálník, but they were not used in this project.

Chapter 2

EEG and Epilepsy

The journal Epilepsia introduced, in its ILAE Official Report [8], a conceptual definition for epilepsy, stating that it is

> a disorder of the brain characterized by an enduring predisposition to generate epileptic seizures, and by the neurobiological, cognitive, psychological, and social consequences of this condition. The definition of epilepsy requires the occurrence of at least one epileptic seizure.

Whereas seizure was defined in the same paper as

> a transient occurrence of signs and/or symptoms due to abnormal excessive or synchronous neuronal activity in the brain.

Usually, epilepsy can be treated with either one or a combination of more anti-seizure medication currently available on the market. Despite the high number of these agents currently available (>25 according to the journal Epilepsia [30]) a significant portion of the patients with epilepsy seem to not respond to any kind of medication. As stated in Chapter 1, a possible solution for these people is to undergo surgery to remove the epileptogenic zone which is causing seizures.

2.1 Inside the brain

Neurons inside the brain communicate through electrical impulses. When a neuron sends an impulse, it creates a certain amount of electricity of which the magnitude determines the strength of the impulse. This strength, also referred to as **voltage** or **amplitude** is measured in microvolts. Neurons of a healthy brain are generating impulses with the voltage in the range of $0 - 200\mu V$ when measured on the scalp [32]. In the terms of **frequency**, its value represents how many times did neuron send an impulse to other neurons. The electrical activity of neurons can be measured by a non-invasive electroencephalograph (EEG). Various EEG electrodes are placed at standardized locations on the patient's head. Such electrodes are displayed in figure 2.1.

Scalp EEG is capturing a summarized activity of big assemblies of neurons on the surface of the patient's head. Based on frequency, we can classify brain waves as *alpha*, *beta*, *delta*, *theta* and *gamma*. These categories are described in detail in table 2.1. In general,

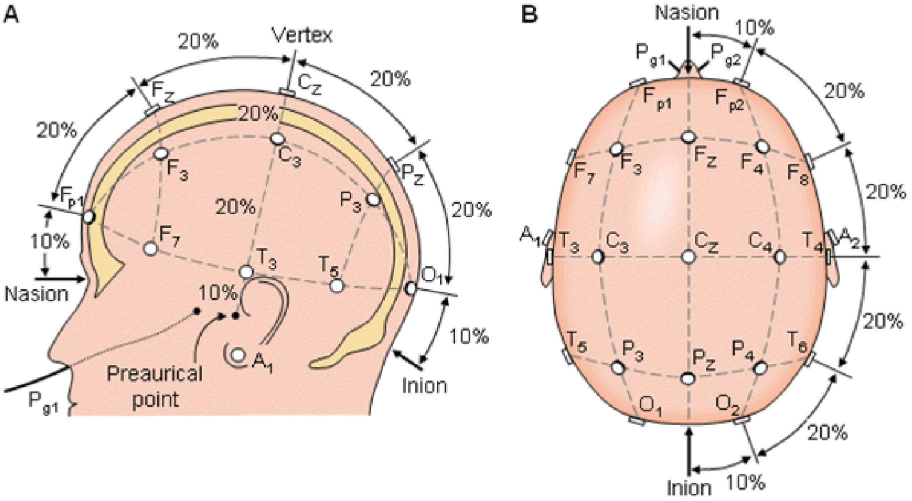


Figure 2.1: Scalp EEG [25].

electrophysiological signals have $1/f$ characterisation, meaning, the higher the frequency, the lower the amplitude. In the context of the amount of information that is being processed in a given state, the more awake the state is, the more information is being processed at once, which results in the higher frequency of brain waves (beta, alpha), whereas the strength of impulses is lower. The opposite applies to the relaxed states (delta, theta).

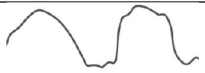
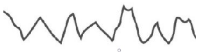
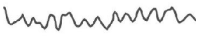
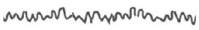
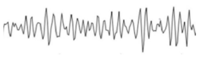
Type	**Frequency**	**State**	Figure
Delta	0-4 Hz	deep sleep	
Theta	4-8 HZ	deep relaxation, REM sleep	
Alpha	8-13 Hz	day dreaming, calm	
Beta	13-30 Hz	alert, active thinking, anxiety	
Gamma	30-100 Hz	high levels of thought and focus	

Table 2.1: Classification of brain waves based on frequency [22].

Classic scalp EEG is used for the diagnosis of epilepsy by capturing the electrical activity of the brain and revealing abnormal patterns compared to its normal function [17]. Results are represented on a graph as signals assigned to a particular EEG channel. Channels are created as the subtraction of the voltage on the reference electrode from an input electrode. Book *Practical approach to electroencephalography* [24] has a whole chapter dedicated to problematics of choosing a reference electrode. Finding a good reference electrode is a difficult task – it cannot be connected it to the ground due to the noise from other electrical devices in hospitals. At the same time, the input electrode would always be a subject of “body noise“ (eye movement, muscle movement, ECG. . .), which would not be present on a ground electrode and therefore would appear on the graph. Part of the body would be a better choice because the body noises from both of the electrodes would cancel out each other. However, placing electrodes too close to each other also poses certain risks. If there was a brain activity common for the reference electrode and for the observed electrode, this activity would be canceled out during the subtraction which would result in a flat signal. Hence placing a reference electrode is a compromise between a location that would cancel out most of the noise, yet would not cancel too much of the brain activity. In general practice, the most commonly used electrode montages are so-called *Bipolar Montages* where electrodes are connected sequentially. The channels then result from the subtraction of neighboring electrodes. Another commonly used reference is an average signal calculated from all EEG contacts. This average signal is then subtracted from individual signals [33].

Unfortunately, epileptogenic zone (EZ) often lies deep within the brain tissue, which means that epileptogenic signals may not reach the scalp, thus the non-invasive measures are usually not adequate for precise localization of seizure onset zone (SOZ) – part of the brain marked for further resection (ideally the same part as EZ). Therefore, implanting electrodes either deep inside the brain tissue or, on the brain surface is necessary. A standard invasive method in clinical routine is an *intracranial electroencephalogram*, commonly referred to as an iEEG. Moreover, iEEG recordings are subject to artifacts caused by muscle and eye movement to a far lesser extent than normal scalp EEG.

2.2 iEEG

EEG recordings obtained in an invasive way are referred to as an *intracranial EEG* or *iEEG*. An article by *Josef Parvizi* [29] provides a good overview of the intracranial EEG, which is summarized in this section. There are two main categories of iEEG (see figure 2.2): Those recorded with the strips or grids of electrodes implanted on the brain surface (i. e. subdural space), and the ones recorded using electrodes penetrating the brain and targeting its specific, pre-defined parts (depth electrode is displayed in figure 2.3). The data from the latter method are the subject of interest of St Anne's Hospital Brno research, same as of this thesis.

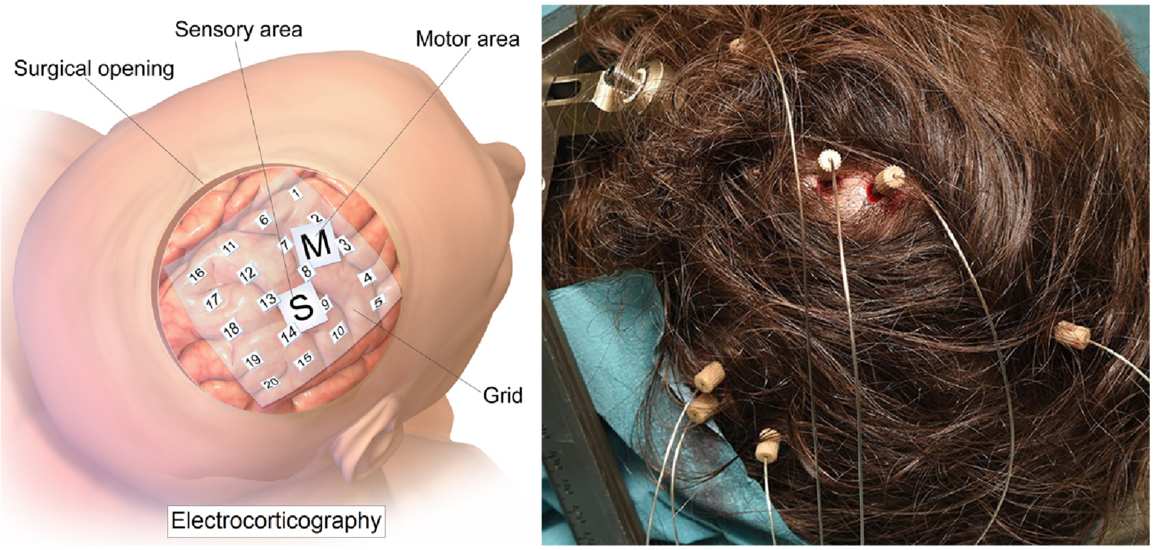


Figure 2.2: Two types of iEEG: *Electrocorticography* (EoCG) [6] on the left, using grids of electrodes, and *stereotactic electroencephalogaphy* (sEEG) [12] on the right, using depth electrodes penetrating the brain.

iEEG plays an important role in the pre-surgical evaluation of epilepsy patients. Yet before the patient is implanted, a hypothesis about the approximate origin of his seizures must be formed with the use of non-invasive diagnostic tools. That implies that only patients with focal epilepsy (i. e. seizures arising from a specific part of the brain) are suitable for invasive screening. Therefore, iEEG is only chosen if the clinicians have high confidence about this condition (referred to as focality), and the preoperative work suggests a high chance of finding the SOZ. Usually, 5-15 depth electrodes are implanted, each consisting of 10-14 recording contacts as displayed in figure 2.3. With these electrodes implanted, the patient is admitted to the operating room, subjected to continuous recording and streaming of raw electrophysiological data from the electrodes. Often, several seizures have to be captured to determine their source. This process, as mentioned in Chapter 1, can take up to two weeks. When comparing classical scalp EEG and iEEG, probably the most notable difference is the resolution. A study comparing the two [2] proved that a signal-to-noise ratio of iEEG could be as high as 100 times higher than the scalp EEG. That can be partially caused by the amplitude of iEEG, which is 10× higher than scalp EEG, and partially because of significantly reduced noise from the recording room, and the physiological noise from eye blinks, eye movements, heartbeat, and muscle movements. Another important differentiator of scalp EEG and iEEG is the sampling frequency used for the signal recording. Scalp EEG is usually recorded, taking into account the Nyquist theorem, with the frequency of hundreds Hz, whereas iEEG with thousands. For example, the testing signal we were processing throughout the project development was sampled at 5000Hz.

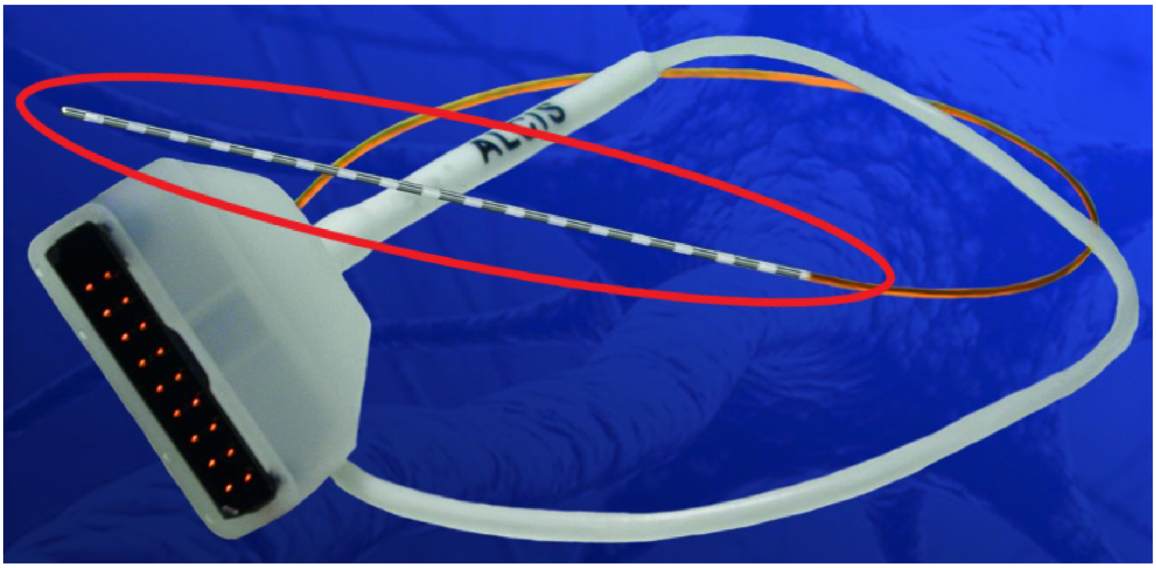


Figure 2.3: An electrode used for iEEG measures at St. Anne's Hospital Brno. Part implanted inside the brain tissue is marked with a red oval. Each metal plate is a contact for the recording of one channel.

2.3 (i)EEG signal processing

The first stage of processing EEG signals, as displayed in figure 2.4, is the removal of artifacts. Several types of artifacts may occur in the EEG recording. Those which fall into the *extrinsic* category are easier to detect and filter out. Extrinsic artifacts are caused by external sources, such as electrode misplacement and cable movements. These artifacts can be eliminated, at least to a certain extent, by strict recording procedures and a cautious approach during the recording. In some cases, the whole part of the signal, which is faulty, is removed. Electromagnetic interference emitted from surroundings is another external source of artifacts in the EEG signal. Artifacts caused by this interference can be easily filtered out because of their distinguishable frequency band using a band-pass filter. A bigger problem is *intrinsic* artifacts that occur due to eye movements, eye blinks, muscle activity, and cardiac activity. Although there is no consensus on an ideal method for removing all types of artifacts, regression methods are the golden standard in this domain. Regression analysis uses a reference channel to estimate the relation between the amplitude of the reference channel and the amplitude of an EEG channel and then subtracts estimated artifacts from EEG. All types of artifacts, as well as artifact removal methods, are described in depth in [16]. The pipeline used at St Anne's Hospital Brno contains a method for labeling faulty parts of the signal, which are not further analyzed.

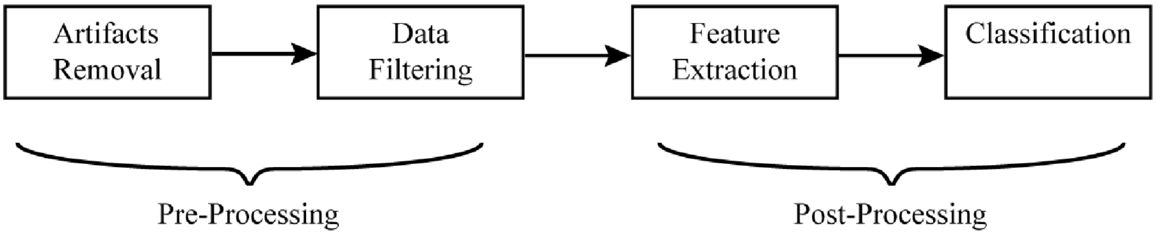


Figure 2.4: Typical EEG processing pipeline [23]

A standard method of analyzing signals – time-frequency analysis can be fruitful in the domain of biomedical signals, particularly EEG signals. At the same time, it can also be incomplete for EEGs. The book *Advanced Biosignal Processing* [26] discusses problems of such analysis. Typical EEG signal consists of multiple frequency components of different magnitudes competing with each other over time, whereas time-frequency analysis only focuses on detecting time-varying spectral power. This analysis may also ignore changes in oscillations and thus fail to provide an accurate description of evolving oscillations. Another problem is transitions in mental state and reactions to external stimuli, which are difficult to track and do not always appear clearly in time-frequency representations. The shortcomings of time-frequency analysis are displayed in figure 2.5. Although we can observe an increase of power in the time-frequency domain (fourth row), it does not say anything about the appearance of the oscillation we are trying to observe. The solution for this is to decompose the EEG signal into separate frequency bands, as shown on the second and third rows of figure 2.5 and to perform time-frequency analysis on each of them separately. However, this approach also comes with a drawback. Cut-off frequencies of every bandpass filter are assumed to remain constant during the whole neurophysiological process under investigation, which can be misleading when the oscillatory component is crossing the cut-off frequency limit. This issue is discussed more in detail in the book [26].

After the artifact removal and data filtering, the post-processing stage follows in the typical EEG processing pipeline. During the feature extraction phase, relevant EEG features are extracted from the signal and later used to derive target observations in the classification stage.

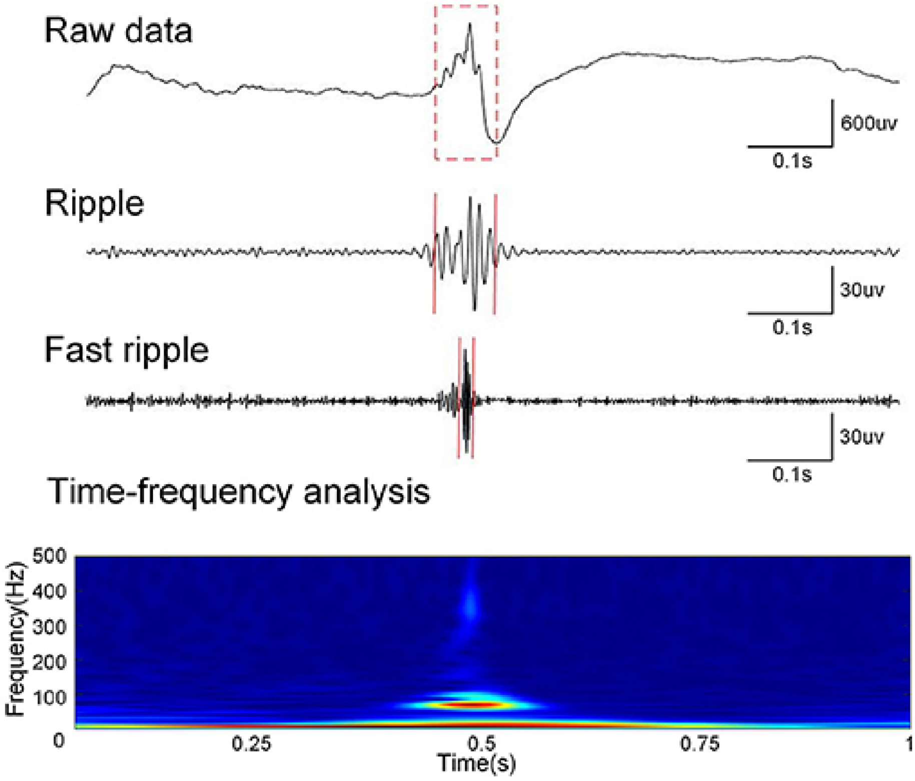


Figure 2.5: Time-frequency analysis (fourth row) applied on raw EEG signal (first row). Although change in the power at 0.5s is observable, the nature of the oscillations cannot be seen. Second row: one second of 80–200Hz filtered data. Third row: one second of 200–500Hz filtered data. Fourth row: Time-frequency analysis of raw data [34].

2.4 Localization of epileptogenic tissue

Since epileptic surgery is a major and irreversible intervention in the patient's brain, accurate localization of the SOZ must be ensured. The clinical gold standard for the SOZ localization and outcome prognostication is based on recording of epileptic seizures and approximation of their origin and spread. To obtain enough data, it takes up to 4 weeks of invasive EEG (iEEG) recordings with reduced patient's medication, which might cause additional side effects, risks and unnecessary costs. The clinical gold standard has not significantly changed over the last 50 years [4]. However, the field of iEEG signal processing has undergone rapid development in recent decades, providing novel, interictal (seizure-independent) iEEG biomarkers of the SOZ, which can improve and significantly accelerate the presurgical evaluation.

The most commonly used interictal markers are *Interictal epileptiform discharges*, also known as *spikes*, and *high frequency oscillations*. **Spikes** are clinically defined as sharp jumps in the signal, with the duration between 20 and 70 ms, that are easy to differentiate from the background activity. In the frequency domain, spike is a local energy increase in the 14.3 – 50 Hz frequency band [15]. A study made by joined forces of Czech and American researchers [15] has proposed an automatic evaluation of spikes as a necessity in the field, since visual analysis of the long-term signals, coming from dozens of channels, is extremely error-prone. Especially when it comes to intracranial EEGs, which come in massive amounts of data, manual analysis is almost impossible.

Another promising interictal biomarker of the epileptogenic zone is EEG activity in higher frequencies, also called **high-frequency oscillations** (HFO: 65 - 600 Hz). Jan Cimbálník in cooperation with researchers from the Mayo Clinic has published a review of current evidence on the interictal (between seizures) HFO and their association with the epileptic brain [5]. According to this paper, HFO are strong evidence of epileptogenic tissues, yet not sufficient to recommend a single patient resection, since they can occur in a healthy brain as well [20]. This fact raises the need to differentiate between their pathological and normal form. Review further states that most of the time, HFOs were shown to be a reliable marker of SOZ only in a group analysis of data summed across SOZ of all patients. Another study found that epileptic cortex may demonstrate atypical cross-frequency interactions [14]. According to the study findings, ictal (during seizure) modulation of pathological HFO (pHFO) by the phase of slow oscillations during seizures is concentrated in the epileptogenic cortex. Slow rhythms happen to co-exist with pathological HFOs and this relationship can be evaluated by computing *modulation index* (MI), in order to assess the strength of the cross-frequency coupling. The stronger the relationship between amplitude and frequency, the higher probability of the SOZ occurrence.

A relatively novel approach is to measure interactions between different frequency bands of the EEG signal. Montreal Neurological Institute conducted a study in 2016 which evaluated a particular form of these interactions – *Phase-amplitude coupling* (PAC), in patients with focal epilepsy during different stages of sleep [1]. Results have shown that PAC between high and low-frequency bands was stronger in the SOZ regions than in the healthy ones. PAC may thus play a significant role in the localization of SOZ.

Jan Cimbálník's study of multi-feature SOZ localization [4] has proven that using several biomarkers to train a support vector machine (SVM) classification is superior to using a single one. The study further argues that due to the nature of brain electrophysiological activity, which is hardly consistent within a time, no single biomarker can be used to effectively localize SOZ in the majority of patients. There are EEG features, whose combination can improve overall SOZ localization by SVM. The study evaluated several of these features. Among the best performing ones was *Local Field Potential*, which is a good indicator of electrophysiological activity. It is also used for studying brain waves during the high-level cognitive functions of a healthy brain (memory, decision making, etc.) [19]. Another one was *Relative entropy*, which evaluates the randomness and spectral richness of two signals, determining how much the entropy of one diverges from the other one. This high performance of entropy feature can be explained by the statistically higher occurrence of spectrally rich events (such as HFO or spikes) in the epileptogenic brain tissue, compared to a healthy one.

As we can see, there are several options for analyzing epileptic signals and localizing pathological tissues in the brain. Novelty of some approaches and the lack of unified methods and codes have been the main drivers for starting the project of which this thesis is a part. Even though automated approaches perform well in the studies, the golden standard remains (and probably will remain for some time yet) identifying SOZ by neurologist over using purely machine-based localization.

2.5 Seizure prediction

One of the biggest hurdles of living life with epilepsy is the unpredictability of seizures. With the knowledge of incoming seizures, patients would be able to take precautions to avoid injuries. Accurate prediction of seizures would also enable a novel approach to epilepsy treatment based on the control and prevention of seizures. Review from 2018 [21] evaluates the current state of the art in this area. We know that seizures are not random and have been shown to have short-range and long-range temporal dependencies. Most of the time, they occur during the rising phase of multi-day rhythms of interictal epileptic activity. However, the fact that there is no predictive characteristic or biomarker that could be used to predict the seizure and would be universal throughout all the patients makes seizure prediction a complicated domain.

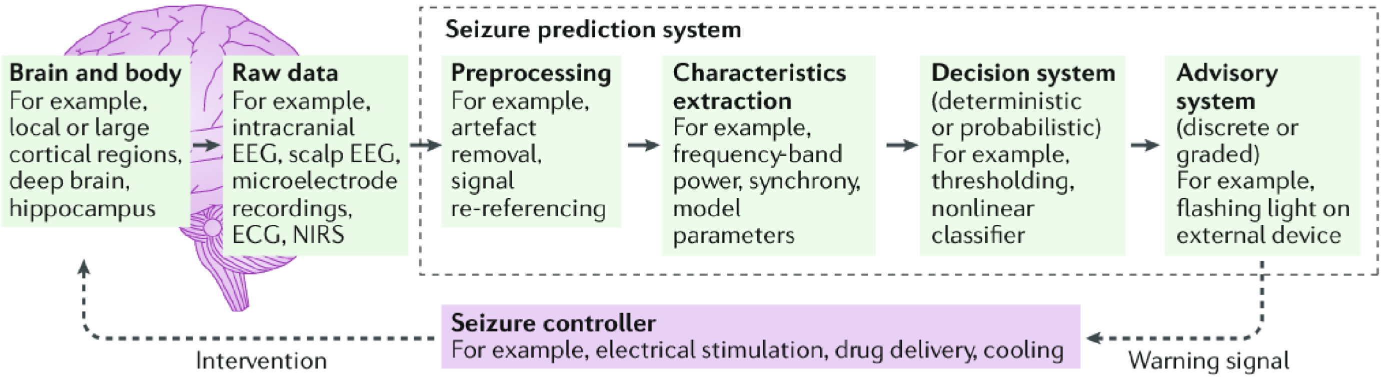

Seizure prediction system
Brain and body
For example, local or large cortical regions, deep brain, hippocampus
Raw data
For example, intracranial EEG, scalp EEG, microelectrode recordings, ECG, NIRS
Preprocessing
For example, artefact removal, signal re-referencing
Characteristics extraction
For example, frequency-band power, synchrony, model parameters
Decision system
(deterministic or probabilistic)
For example, thresholding, nonlinear classifier
Advisory system
(discrete or graded)
For example, flashing light on external device
Seizure controller
For example, electrical stimulation, drug delivery, cooling
Intervention
Warning signal

Figure 2.6: An illustration of how seizure prediction system pipeline might look like [21].

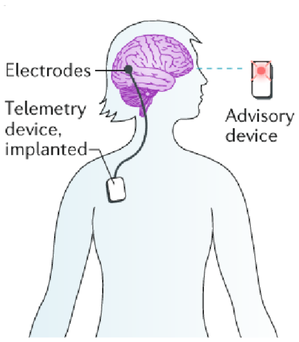


Figure 2.7: Seizure prediction system illustrated in 2018 seizure prediction review [21].

Merely one clinical trial with a fully functioning prediction system has been conducted so far [7]. The prediction system was based on intracranial electrodes connected to a telemetry unit that was sending data to the device. That was processing them in real-time and displaying warning lights to the implanted patient when the seizure was impeding (prediction system is displayed in figure 2.7). It was portable, and patients participating in the study were implanted with it for several months. Seizure prediction was excellent for three out of nine patients. This study brings a new light to the discipline by demonstrating that long-term recordings are possible and that seizure prediction is achievable, yet patient-specific. The tool that we are implementing in this thesis highly overlaps with this domain since real-time signal processing is a significant part of successful seizure prediction and might be of use in future research, that is eagerly awaited by patients with epilepsy.

Chapter 3

Data

We were working with the intracranial EEG data recorded at St Anne's Hospital Brno throughout the project development. Thanks to the treatment of the number of patients with epilepsy and multiple studies conducted in this institution, we were able to work with a data set of iEEG signals with labeled pathological channels and available results of computed biomarkers. Data sets from four patients were used for a simulation of real-time data flow, flowing into our implemented tool. All the data sets we used during the development and testing were anonymized, and patients were assigned with IDs.

3.1 Short data

We used a ~30 min of relaxed stated iEEG recording to continuously test the functionality and performance of our tool throughout its development. The data set was recorded by a 150-channel research iEEG acquisition system (M&I; Brainscope, Czech Republic). The sampling rate was 25 kHz during the recording. The iEEGs were then low-pass filtered and down-sampled to 5 kHz for further processing. This sampling rate gives all-together $9,491,505$ samples for a single channel.

Such short recordings are made while the patient is lying on the bed without exerting any activity. This approach may be troublesome since the recordings may not always be accurate. As the latest research suggests [18], the epileptic brain is active in cycles occurring in different timescales, such as circadian (~24 h cycle), multidien (cycles lasting >2 days up to several weeks), and circannual (1-year cycles). There is a high probability that during the selected 30 min window, the electrophysiological activity of the epileptogenic focus will be low. Due to this fact, much larger data sets were used for the evaluation of the finished tool (see next section). The data were flowing to the tool in 5 s segments. Results computed offline:*relative entropy* and *signal statistics* were available as a reference to the data computed in 'real-time'.

3.2 Long data

Larger data sets were used for the final evaluation of the tool. The recordings were obtained during the patient's routine activities, such as eating, using the phone, or walking around the hospital. All patients were implanted with depth electrodes as part of their pre-surgical analysis for the treatment of pharmacoresistant focal epilepsy. Used electrodes were either DIXI or ALCIS (diameter = $0.8\,mm$; inter-contact distance = $1.5\,mm$, contact surface area = $5\,mm^2$; contact length = $2\,mm$). The acquired iEEG was low-pass filtered and down-sampled from 25 kHz to 1,000 Hz for subsequent storage and analysis. The used recording reference was the average of all intracranial signals.

3.2.1 Patient 95

The data set of patient 95 consisted of 172 channels and was ~13 hours long. Channels **P'6** and **P'7** were labeled as pathological by a neurologist, while channels **P'1** to **P'9** were all exerting signs of epileptogenic activity based on the values of relative entropy and detected HFO computed during the offline procedure. For the evaluation, real-time computed data were pulled out of the database and separated into a segment of 30 min length. Channels without offline computed reference results were excluded from the assessment. Furthermore, channels that were out of the patient's head during the recording were excluded as well.

3.2.2 Additional patients

The tests described in chapter 6 were performed on two additional patients. The results of those patients are available in appendix A.

Patient 83

The data set of patient 83 consisted of 165 channels and was ~5 hours long. Channels L1 and L2 were labeled as pathological by neurologists. Channels without offline computed reference results were excluded from the assessment. The relaxed iEEG of patient 83 without seizures is shown in figure 3.1. iEEG with an ongoing seizure on channels L2 and L3f from the same patient is displayed in figure 3.2.

Patient 79

This data set was recorded with a 68-channel research iEEG acquisition system and the recording was ~4 hours long. Channels B'1, B'2, and B'3 were labeled as pathological by neurologists. Channels without offline computed reference results were excluded from the assessment.

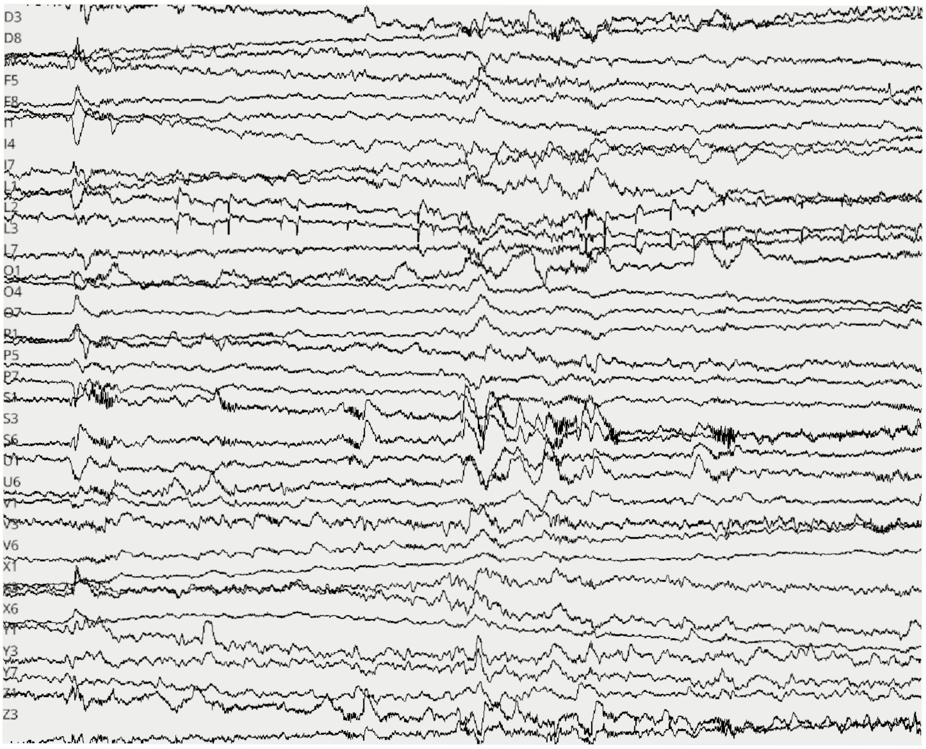


Figure 3.1: A 30 s (axis x) of seizure-free iEEG record from patient 83. Axis y displays iEEG channels.

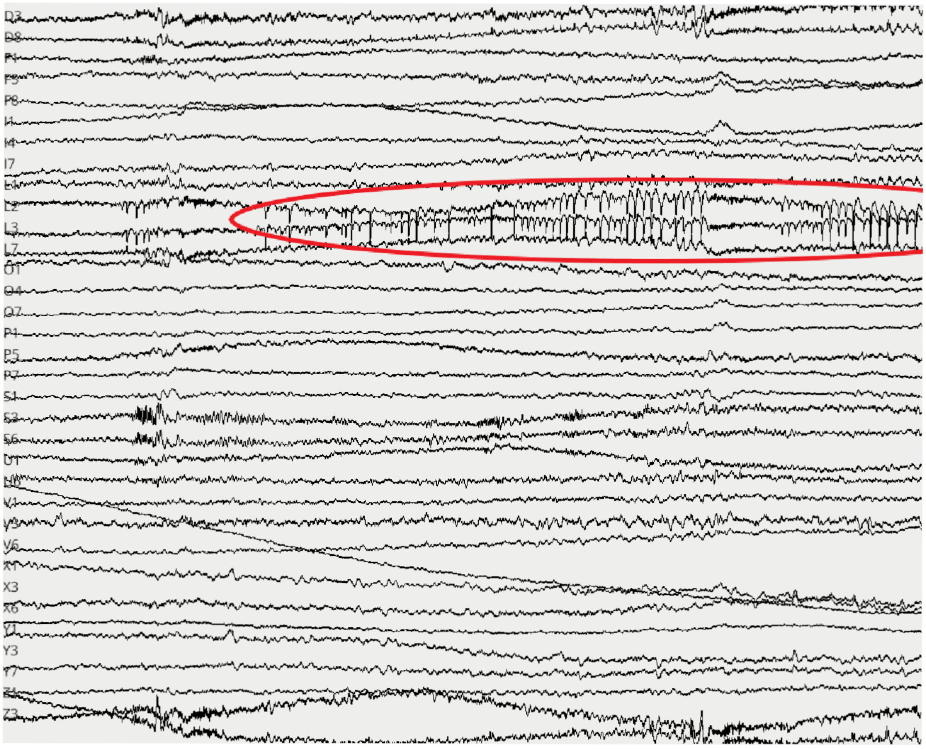


Figure 3.2: A 30 s (axis x) iEEG record with captured seizure from patient 83. Axis y displays iEEG channels. Seizure is visible on channels L2 and L3 (red oval).

Chapter 4

Tools and algorithms

This chapter describes the three libraries, on which the whole project is based, together with some other tools used for the implementation.

4.1 Epycom library

When I first started collaborating with BME ICRC at St. Anne's University Hospital in Brno, I joined the work on the python library named Epycom[1] – ElectroPhYsiology COmputational Module, which was built considering novel approaches towards localization of epileptogenic tissues combined with the traditional methods and algorithms. It implements many of the promising iEEG biomarkers, such as phase-amplitude coupling [1]. Overall, the package provides tools for the computation of various features of iEEG signals that can be further analyzed by artificial intelligence or evaluated by physicians.

Epycom consists mainly of the three types of functions – *univariate*, *bivariate* and *event detection*. The main differentiators between them are the input, output and nature of the computation, which make a given function to fall into a particular category.

4.1.1 Univariate functions

These functions take as an input time series from a single channel and are usually returning a single `float` value. As an example, we can mention the function for computing mean vector length (MVL), which returns a value assessing the strength of the cross-frequency coupling. I chose to describe this algorithm here because I implemented it as a part of Epycom library prior to the start of this project. MVL is computed by averaging complex numbers obtained from combining phase and amplitude extracted from different frequencies of an analytic signal. Analytic signal is a complex-valued function that consists of the original function as the real part and its Hilbert transform as the imaginary part[2]. Hilbert transform can be understood as a phase shift of every frequency component of a function by $+\frac{\pi}{2}$ for negative frequencies and by $-\frac{\pi}{2}$ for positive frequencies[3]. Therefore, the analytic signal of $x[t]$ can be mathematicaly written as:

$$x_a(t) = x(t) + i\Big[\frac{1}{\pi t} * x(t)\Big],$$

[1] `https://gitlab.com/icrc-bme/epycom`

[2] `https://en.wikipedia.org/wiki/Analytic_signal`

[3] `https://en.wikipedia.org/wiki/Hilbert_transform`

where $\left[\frac{1}{\pi t} * x(t)\right]$ is the Hilbert transform and sign $*$ means convolution. Extracting the amplitude from high-frequency filtered analytic signal and phase from the low-frequency filtered analytic signal, we can compute mean vector length in the discrete domain as:

$$MVL = \left|\frac{\sum_{n=1}^{m} a_n e^{i\theta_n}}{m}\right|,$$

m is here the total number of data samples, a_n is an amplitude extracted from high-frequency filtered analytic signal at data point n and θ_n is phase extracted from low-frequency filtered analytic signal at data point n. Complex numbers represent vectors in a polar plane. The result is a mean vector, whose length resembles the amount of phase-amplitude coupling. If there is no phase-amplitude coupling present, all vectors cancel out and the mean vector will be short, otherwise the vector will be of a significant length (figure 4.1).

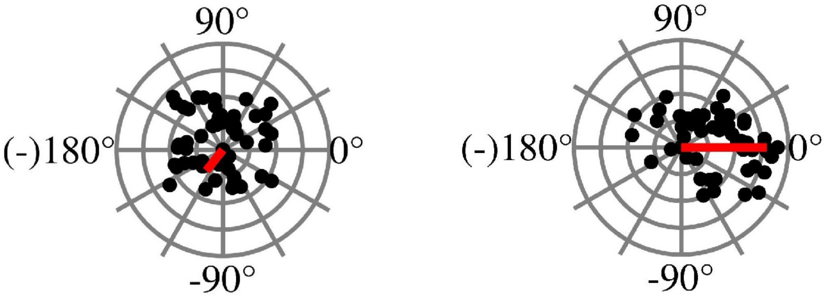


Figure 4.1: Illustration of mean vector length method used to asses phase-amplitude coupling of the signal [13]. Signal, with no coupling is displayed on the left, signal with high degree of coupling displayed on the right.

4.1.2 Bivariate functions

Functions from the Bivariate category take as input signals from the two adjacent channels and compute relationships between them. An example from this category can be function for computing relative entropy (REN) between the two signals. This function was also used as a reference function for the evaluation of computations of our real-time EEG processing tool. Relative entropy is computed as a Kullback-Leibler (KL) divergence of two input signals. In theory, KL divergence is computed from two probability distributions. If $p(x)$ and $q(x)$ are the probability distributions of a random variable x, KL divergence is computed as:

$$D_{KL}(p(x)||q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)},$$

In the Epycom function `compute_relative_entropy()`, histograms $h1$ and $h2$ are created to represent probability distribution of each of the two input signals across 10 equally wide bins. In the next step, KL divergence of the histograms is computed from both sides ($h1$ with respect to $h2$, and $h2$ with respect to $h1$). The higher estimated entropy is returned.

4.1.3 Events

Functions performing the **event detection** are the most complex. Two types of events are implemented in Epycom – *spikes* and *HFOs*. They are the most commonly used epilepsy biomarkers, both are discussed in Chapter 2. Several algorithms for detecting HFOs are implemented here, as well as the function for detecting spikes. The input is a single signal, whereas the output is a tuple of several values containing information about the given event, such as event start, event stop, duration, low and high frequencies, etc. Output information may vary with the used algorithm. From this category, we used implementation of an algorithm for detecting HFOs, referred to as CS (Cimbálník-Stead) detector [3] for validation of our real-time processing tool. This algorithm was designed with to respect to efficiency to be suitable for real-time processing. Explaining in details is beyond the scope of this thesis, but interested reader can read more in [3].

4.1.4 Organization of library

Each epycom function is wrapped in a class of the same name, inheriting from the class `Method` implemented in this library. This class stores metadata about the particular function, such as `algorithm`, `algorithm_type`, or `version`. Moreover, class `Method` is implementing methods for running the Epycom function stored as its class variable or running the given function using a sliding window, where the analyzed signal is divided into windows of a given size. The latter method, named `run_windowed()` is also used in the implementation of `EpycomNode` in the Mepior library discussed further.

In this thesis, we were dealing with the challenge of the EEG signals delivery into the interface, where multiple functions from the Epycom library can be applied to the data at the same time. We created an infrastructure that works with signals in real-time, computes their parameters and epileptic markers (using Epycom library), and stores them in a database, from which they can be further extracted and processed. After the tool is tested and evaluated, a server-based database will be used for storing the data. But during the time of developing a tool for real-time signal processing, we were sending the data to a locally created database. The details about the storing of data are described in the Implementation section of Chapter 5.3.

4.2 Pyacq

The foundation of our project is Pyacq [4] – a python library implementing an API for the processing of data streams. It provides an interface consisting of several types of nodes, allowing to build data processing pipelines. Nodes can interact with devices, generate data, store data, perform computations, or display user interfaces and send the data further in a form of the output stream. Each node can have multiple inputs and outputs connected to the other nodes.

The library is built in an object-oriented way, where each node is implemented as an individual class inheriting from the abstract class `Node`. This class contains methods common for all the nodes, such as `configure()`, `initialize()`, `start()`, or `stop()`, which are reimplemented by the subclass based on its specification. The `Node` class has also properties `input` and `output`, which are instances of classes `InputStream` and `OutputStream`

[4] `https://github.com/pyacq/pyacq`

and both are instantiated when a particular node is created. They allow nodes (which may exist on different threads, processes, or machines) to send and receive data to and from other nodes. Schema of data flow in one instance of a node is displayed in figure 4.2.

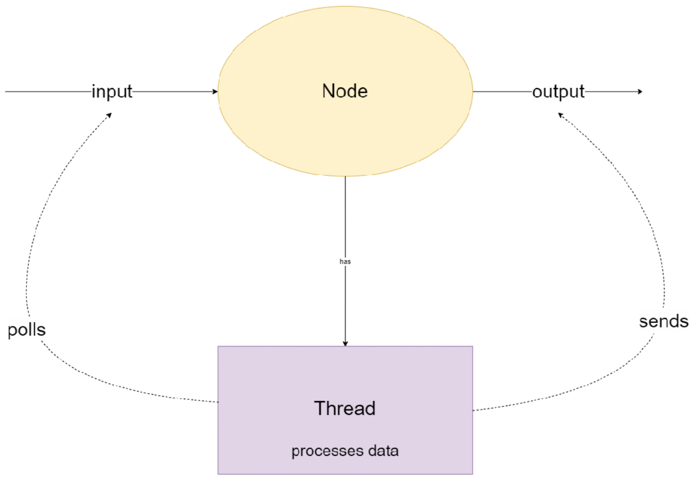


Figure 4.2: Schema of relationships between the individua Pyacq classes

Most of the nodes create their own thread, stored as their class variable, instantiated when a method `initialize()` is called from an instance of a node. The thread is an instance of a class implemented exclusively as a thread for the given type of node. Thread classes are usually inheriting from the another abstract class `ThreadPollInput`. The purpose of node thread is polling an InputStream associated to that node in the background and emitting a signal when data is received. `ThreadPollInput` contains method `process_data()` that is called from the polling thread when a new data chunk has been received. By default, this method only emits the signal `new_data` with the updated stream position, but most of the time it is overridden and data are being processed here and sent further. Figure 4.3 illustrates example of class diagram for node `ChunkResizer` with selected parameters and methods.

Data between nodes are transported through `zmq.PUB` sockets either directly or via shared memory when only the index of a current frame is sent in the socket[5]. The great advantage of pyacq lies in the ability to use several CPUs as well as remote machines and process data in parallel. The library implements tools for managing multiple processes through the class `Manager`. Single process, that consist of a group of nodes (gathered in the object named `NodeGroup`), can be assigned to a remote machine that is running a host server[6]. It can significantly speed up the computations that are mutually independent and allows us to shorten the time needed for data processing.

Our project has been initiated by forking a Pyacq repository and building upon it our own infrastructure for processing EEG data with the functions from Epycom library.

[5] `https://pyacq.readthedocs.io/en/latest/apiref/core.html`
[6] `https://pyacq.readthedocs.io/en/latest/manager.html`

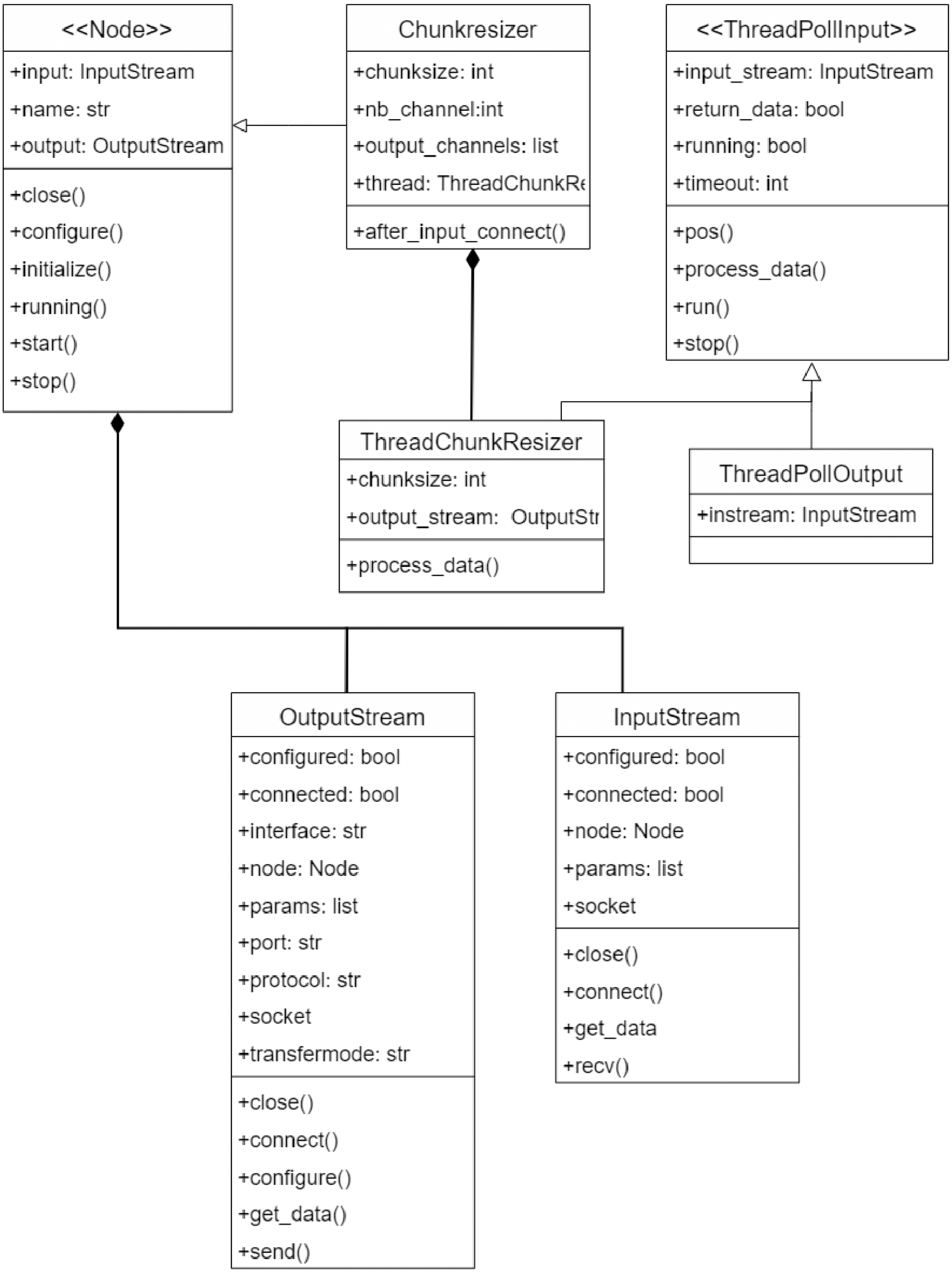


Figure 4.3: Reverse-engineered class diagram of an example Pyacq node `ChunkResizer`. Classes `InputStream` and `OutputStream` are fully dependent on the node, that instantiates them and stores them as its parameters. On contrary, there is a weak reference between the streams and `ThreadChunkResizer` class which means that this reference is not enough to keep the streams alive (even though they are stored as `ThreadChunkResizer` parameters), unless there is other remaining reference (in this case reference to the node). Only selected methods and parameters are displayed on the diagram.

4.3 Mepior

By modifying the Pyacq library we created a new library called Mepior. The nodes not needed for the EEG processing infrastructure were left out, while new nodes have been implemented. These nodes include:

- `EpycomNode` exclusively for applying Epycom functions on the incoming data stream
- `SQLitePusher` node for pushing the data into the database
- `EasysSocket` node for establishing a connection throughout a BSD Socket with the external data stream (using python library `socket`[7])

Each of these nodes, same as the native pyacq nodes, is implemented as a class inheriting from the abstract class `Node`. The operation for which a particular node is responsible (for example, sending data into the database) is conducted within a node thread, a class variable belonging to that node. This mechanism is described in details in section 4.2.

This library was used and specifically designed for building a pipeline that would satisfy the requirements for real-time EEG processing.

4.4 Signal filtering

Pyacq implements node `OverlapFiltfilt` that allows filtering of the incoming signal chunks. This node implements the forward-backward method using a second-order (sos) coefficient and a sliding, overlapping window. Node applies externally provided sos coefficients on incoming data chunks using `scipy.signal` function `sosfilt` [8]. This function implements series of second-order filters with *transposed direct form 2*. One series are computed as:

$$\begin{aligned} y[n] &= b_0 x[n] + s_1[n-1], \\ s_1[n] &= s_2[n-1] + b_1 x[n] - a_1 y[n], \\ s_2[n] &= b_2 x[n] - a_2 y[n], \end{aligned}$$

Here, the a_1, a_2, b_0, b_1 and b_2 are the supplied coefficients, x is the raw signal and y is the new filtered signal. Flow diagram of this filter is displayed in figure 4.4.

The forward-backward method applies the filter twice. First, the filter is applied to the signal (in our case chunk of the signal), then the result is reversed, the filter is applied for the second time, and the result is reversed again. The reason for doing this is a zero phase shift in the filtered signal.

In our pipeline, we used the 3rd order Butterworth bandpass filter for computing the coefficients using `scipy.signal` function `butter` [10]. Although the signal filtering plays a significant role in the iEEG processing and thus in our pipeline as well, during the tool evaluation, filtering node was omitted. This way, we wanted to avoid the possibility of inaccurate computations caused by the differences between the filtered signals and focus

[7] https://docs.python.org/3/library/socket.html

[8] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.sosfilt.html

[9] https://en.wikipedia.org/wiki/Digital_biquad_filter

[10] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.butter.html

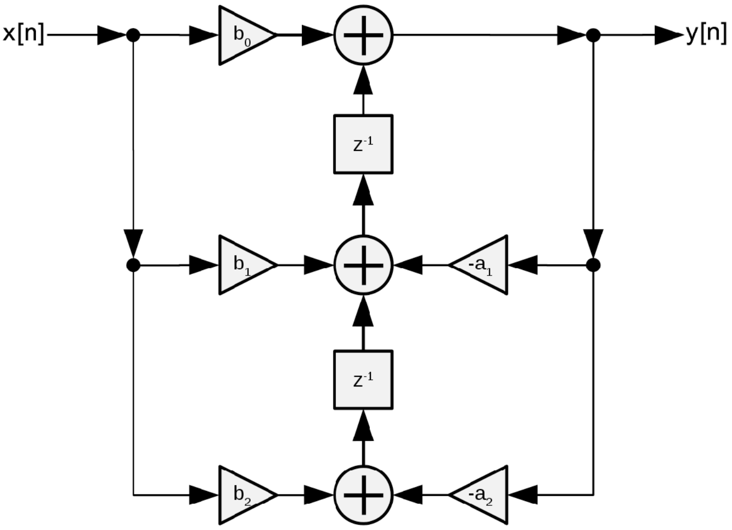


Figure 4.4: Flow diagram of the *transposed direct form 2* of the second-order series filter used in the filtering node[9]

more on computing iEEG biomarkers. Therefore, we used computations from the unfiltered signal as a reference. Moreover, some Epycom methods implement their filters or require signal filtered in particular frequency bands, and the topic of future work will be to implement a robust solution for the signal filtering, that would work for all of the Epycom functions universally.

Chapter 5

Implementation

Before the actual computation of the EEG features, signal has to be divided into chunks and filtered to extract the features that reflect different electrophysiological phenomena occurring in the brain. There is also a question of how and where to store a big amount of computed data. All of these aspects have been taken into account while designing the architecture of our real-time EEG processing tool.

The project is implemented in `Python3`[1], with the use of some additional tools, that are exceeding this language (such as SQL engine).

5.1 What was working

As previously mentioned, my colleagues had initiated the project before the start of my contract, but the project was later postponed until we re-opened it again as my next work assignment. Nevertheless, many things were implemented during the initiation phase. For the first couple of months, I had to study current architecture and proposed solutions. That required reading the `pyacq` documentation and understanding the principles of its classes. Mepior library, described in the previous chapter, was implemented by Jan Cimbálník to a great extent, although class `EpycomNode` was not yet adjusted for processing various types of the Epycom functions, same as class `SqlitePusher`, which is responsible for storing computed data into the database. A program written in C, named *Easys EEG Data Server (eeds)*, was used to simulate a live data stream. It allows sending parts of an EEG recording specified in a configuration file as UDP packets, which are received by the first node from the class `EasysSocket`. This program was configured to send packets of 5000 samples for each of 150 channels from the reference file on localhost. Jan Cimbálník also implemented a prototype of a script with basic architecture of a real-time signal processing tool in a sample script which was used as a baseline for my implementation. This architecture is sketched in Figure 5.1 which also describes the flow of data throughout the pipeline. Such architecture remains very flexible when it comes to adding or removing nodes (for example, we can insert a node from class `MefRecorder` after the filtering node for capturing the filtered signal into the `.mefd` file or visualize the filtered signal through the node from the class `QOscilloscope`).

The data are flowing in the form of `OutputStream`, which is a class implemented in the original pyacq library. The detailed description of the input and output stream can be found in the section 4.2 of Chapter 4.

[1] `https://www.python.org/`

5.2 Architecture

Our pipeline consists of multiple nodes as discussed in the previous sections. The process of creating and initiating node is similar for all cases. First, an instance of a particular node class is created. Then `node.configure()` function is called with parameters specific for the given type of node (for example, for `ChunkResizer` a chunksize is passed as a parameter). If the node input is supposed to be connected to the output of different node, it is ensured by calling `node.input.connect()` function on the instance of that node passing the second node output as a parameter. The output of this node has to be configured as well. That is done by calling function `node.output.configure()` specifying protocol and interface used for sending the data from it in the parameters. Finally, node thread is created by calling `node.initialize()`. Thus, instantiating and configuring node can look like this:

```
# ----- Chunk resizer node -----
chunkresizer = ChunkResizer()
chunkresizer.configure(chunksize=50000)
chunkresizer.input.connect(dev.output) # connect to the output of node dev
chunkresizer.output.configure(protocol='tcp', interface='127.0.0.1')
chunkresizer.initialize()
```

Schema of the pipeline is displayed in figure 5.1. All nodes are configured to use TCP protocol and send the output to localhost interface. The first node is an instance of `EasysSocket`, which is responsible for catching the incoming signal from an external source and sending it further to the instance of a class `ChunkResizer`, which takes a multi-channel input signal stream and ensures that the output is the same packet with the length of a chunksize. Next, the data flows into the `OverlapFiltFilt` node which applies the filter on incoming chunks. After that, output of filtering node can be connected to the input of various nodes computing iEEG biomarkers (instance of class `EpycomNode`). Throughout the development, we used only three such nodes, each carrying an Epycom function of one category. Ideally, most of the Epycom functions will be used in practice, but it is not necessary to implement them all for the prototyping and evaluation. Because if the representative of the particular category proves to be accurate in comparison with the reference offline computations, we may safely assume that the rest of the functions from the same category will also be accurate. That is because the implementation of Epycom function remains the same and our main concerns which may affect the results in real-time processing are phase shift and packet loss. As a representative of univariate category, function `SignalStats` was used (function computes simple statistic about the incoming signal, such as mean amplitude, maximum and minimum amplitude, etc.), from bivariate category function `RelativeEntropy` was used and for event computing we selected function `CSdetector` (functions RelativeEntropy and CSdetector are described in details in chapter 4.1). Each of the Epycom nodes is connected with its output to the instance of class `SqlitePusher`. This class sends the incoming data to the database (details described in the following section).

5.3 Data storing

Initially, the output data was stored in a local database file. Data saving was implemented with the use of python library `sqlalchemy`[2], which creates a SQLite engine, and

[2] `https://www.sqlalchemy.org/`

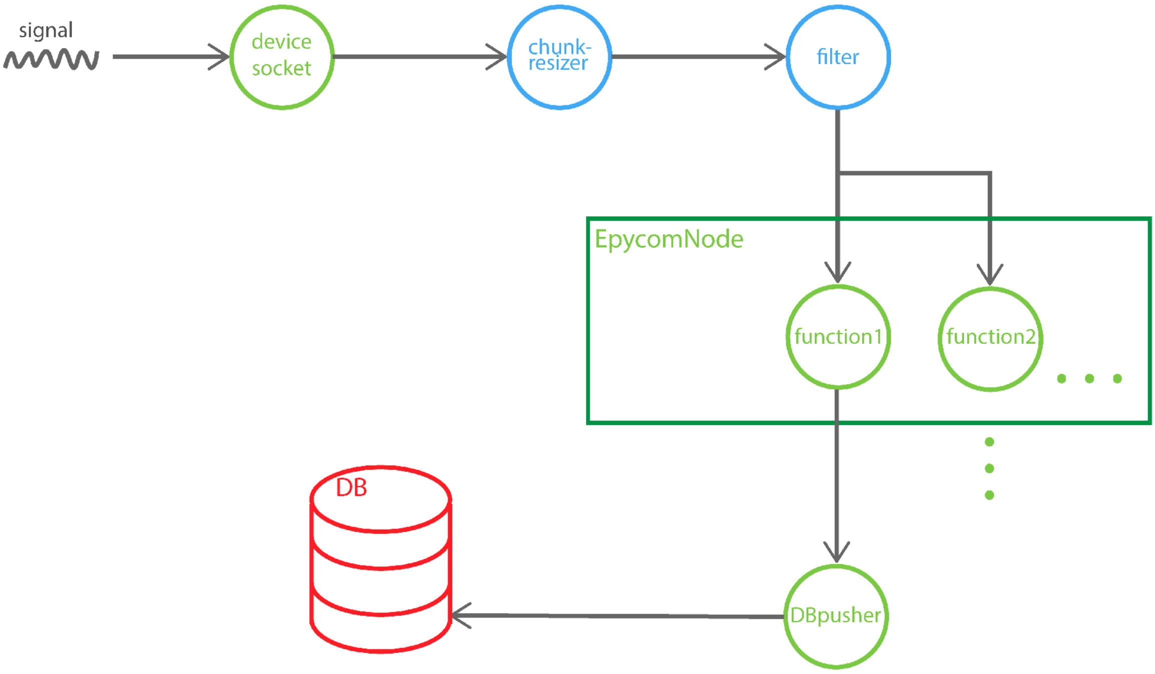


Figure 5.1: Architecture of the protype pipeline for real-time EEG processing. Blue nodes are from the original Pyacq library, green nodes are implemented in the Mepior library.

`pandas.DataFrame`[3], a class that enables pushing its data into the SQL database. The tables with the computed features are created and filled by the respective node from the class `SqlitePusher`. Each `EpycomNode` output stream is assigned to a separate `SqlitePusher` node, which stores the data flowing from it into the table exclusive to the given Epycom function. Additionally to the tables with computed features, the database contains tables with data about the functions, channels, algorithms, etc. ER diagram of these tables is in the figure 5.2.

This solution is sufficient for evaluating the performance and functionality of the tool, yet for practical implementation, a different database must be used. The database server of FNUSA ICRC has been developed in parallel with this project by Ján Címbálník, and in the future, it will serve as a proper data storage for the computed data.

[3] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html

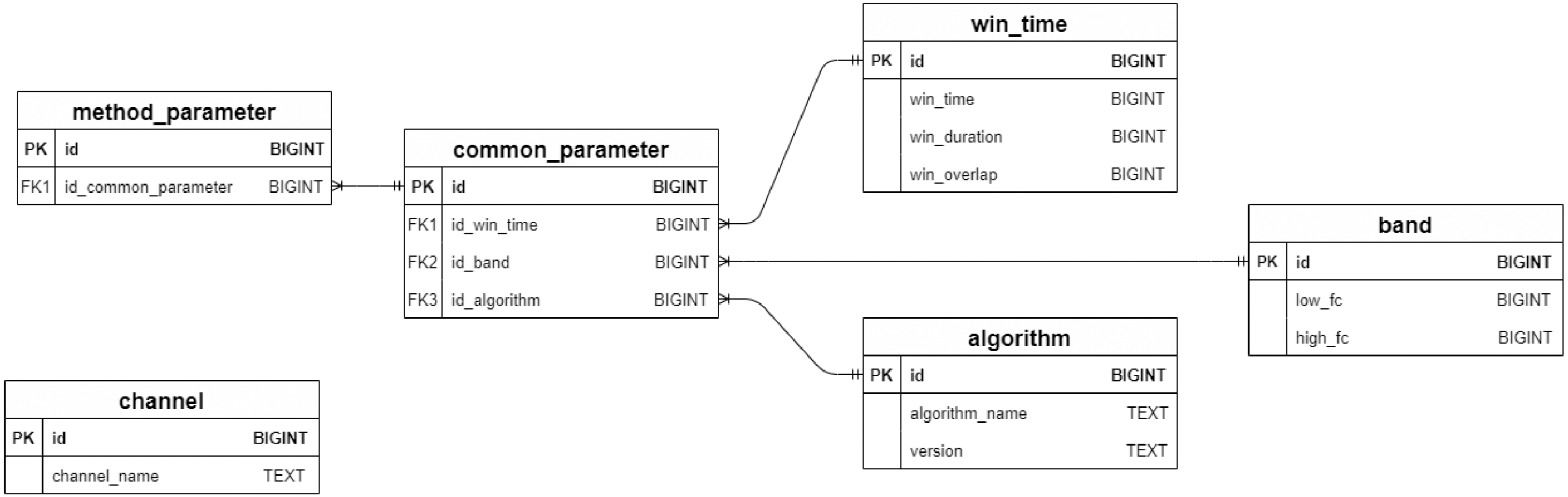


Figure 5.2: ER diagram of the tables with additional data other than those which have been computed by `EpycomNodes` in the temporary local database used during the development. Tables with computed features are not present on this diagram.

5.4 My work

5.4.1 Class EpycomNode

The purpose of this node is to apply the Epycom library to an incoming input data stream. As discussed previously, Epycom function may fall into one of three categories – univariate, bivariate, or event. EpycomNode has also its thread – EpycomNodeThread and Epycom function is applied within the thread method process_data(). I implemented this method, so it applies Epycom function based on its type throughout all the channels, concatenates results into one array and sends it further. As the function loops through the channels, the input data with the shape of *buffer size * number of channels* at the given channel index are passed to the Epycom function in the case of event or univariate Epycom function. When it comes to bivariate functions, data from neighbouring channels (channels at index i and $i+1$) are passed to the function. The algorithm for main part of method process_data() could be written in pseudocode as:

```
for channelIdx in channels do
    if methodType is 'bivariate' then
        if channelIdx is last(channels) then
            $result = epycomFunction(dataAt(channelIdx), dataAt(first(channels)))$
        else
            $result = epycomFunction(dataAt(channelIdx), dataAt(channelIdx + 1))$
    else
        $result = epycomFunction(dataAt(channelIdx))$
    append $result$ to $allResults$
send $allResults$ to $output$
```

The channel names are stored as the class variable of InputStream connected to the EpycomNode. EpycomNode has to be instantiated and configured with the appropriate Epycom function parameters, for example as:

```python
# ----- Epycom node parameters -----
from epycom.univariate import SignalStats
sigStatsParams = EpycomNodeParams( compute_class=SignalStats,
                                win_size=5000,
                                epycom_module='epycom.univariate',
                                epycom_class='SignalStats',
                                db_table='signal_stats')

# ----- Epycom node -----
proc = EpycomNode(**sigStatsParams.get_node_params())
proc.configure(sigStatsParams.win_size)
proc.input.connect(chunkresizer.output)
proc.input.set_buffer(sigStatsParams.win_size)
proc.output.configure(protocol='tcp', interface='127.0.0.1')
proc.initialize()
```

5.4.2 Class `SQLitePusher`

SQLitePusher is responsible for sending the data from its input stream into the database, whereas the input stream of this class is connected to the output stream of the class EpycomNode. Data are being processed and stored in database again from methode `process_data()` belonging to the class `ThreadPush`. Similar to the EpycomNode, I implemented different ways of storing the data, depending on the type of Epycom node function, which is a variable of the input stream. Different EEG features require different information about their data to be stored. Besides the nature of the computed data, the table of the bivariate function contains columns for indexes of two channels. Event and univariate tables have a column with only one channel. At the same time, tables of event functions comprise additional columns `event_start` and `event_stop` with the time of the beginning and end of the computed event. The mechanism of storing the data in the database is: first converting data from the input stream into the `pandas.DataFrame`[4], then adjusting the data frame to the requirements of Epycom function type, and last sending the data into the database by applying function `to_sql()` on a given data frame. In order to do that, SQL engine must be created within the thread. The main part of the algorithm could be written in pseudocode as:

```
if methodType is not 'event' then
    throw away 'event_start', 'event_stop'   from inputData
    if methodType is 'bivariate' then
        map channels to inputData(channelId1)
        map (rotate channels by +1) to inputData(channelId2)
    else
        map channels to inputData(channelId)
send inputData to DB
```

[4] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html

5.4.3 Pipeline implementation

The prototype of a pipeline (displayed in figure 5.1) was partially implemented by Jan Cimbálník. I finished the implementation of the testing pipeline script by implementing the function `create_tables()` and by implementing the three Epycom nodes, each for a function of one of the categories. Function `create_dables()` creates initial tables described in figure 5.2. Table relationships and data consistency is handled manually in this function, since `sqlalchemy` and `pandas` don't implement advanced database operations (after connecting the tool to proper FNUSA database, manual handling will not be needed).

After the testing pipeline was functional, I implemented a distributed pipeline – a high-level implementation for running the original pipeline across multiple processes and possibly using several machines at a time. Here, the non-working prototype of the script was done already by Jan Cimbálník; I brought it into operation and adjusted according to the testing pipeline. Schema of the distributed pipeline is shown in figure 5.3.

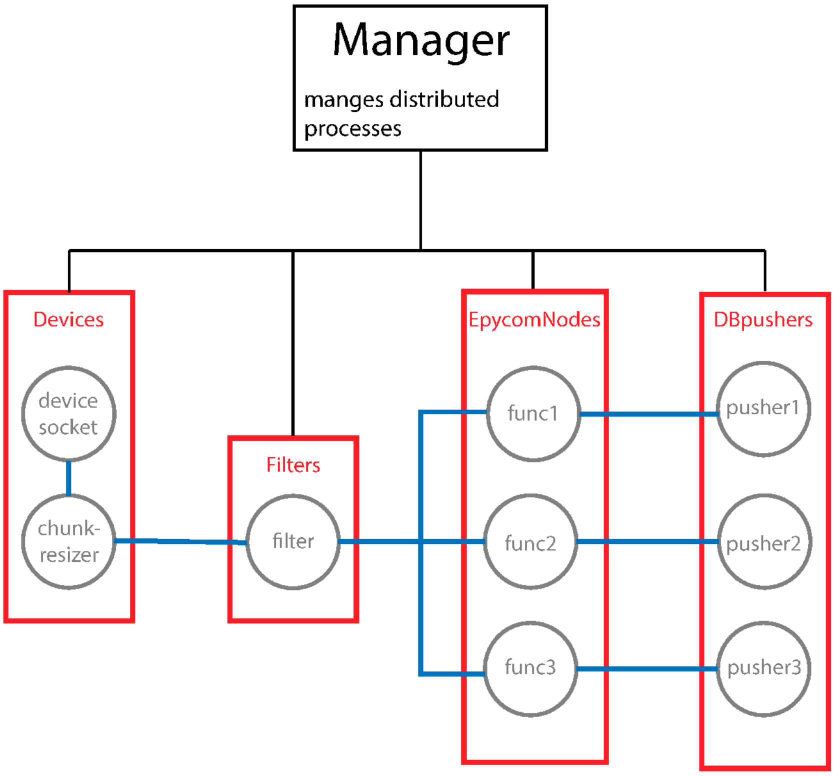


Figure 5.3: Architecture of the pipeline distributed across multiple processes (possibly machines). The central point is the Manager, that is taking care of distributed processes - NodeGroups (symbolized by red squares). NodeGroup can manage several nodes within a single process. Nodes are created within a particular NodeGroup, to which they belong. The instantiating of node is otherwise the same as in local, single-process implementation.

Chapter 6

Testing

To appropriately evaluate our tool, we asked four questions:

Question 1 *Does our tool process the data the same way as when they are processed offline?*

Question 2 *Is the tool robust enough to process long-term signals?*

Question 3 *Does the tool achieve similar performance as the reference 30 min. relaxed-state recording?*

Question 4 *Does the ability to localize SOZ using biomarkers changes over time and can it outperform referential recording in some segments?*

Results of patient 95 recording have been used to evaluate questions 3 and 4 in this chapter. Results calculated from recordings of patient 83 and 79 are available in the appendix A.

6.1 Methods

Question 1

Does our tool process the data same way as when they are processed offline?

A short, ~30 min recording (described in 3.1) was processed, and the results were used to answer the first question. Data from 122 of 150 channels were selected for the evaluation (reference and noisy channels were omitted). Reference results were available for Epycom functions `RelativeEntropy` and `SignalStats`. The latter computes simple statistics about the signal, such as standard deviation of signal power, maximum power of the signal, or mean power of the signal. A mean value of each of these statistics, along with the relative entropy, was computed for every channel in both reference and newly computed datasets. The newly created datasets were then analytically compared.

The paired Student's *t-test*[1] was computed for the pairs of reference and newly calculated results. The *t-test* can be used to estimate whether the means of two sets of data are different from each other. The formula for computing statistics t between two datasets with the same number of samples n is:

$$t = \frac{\overline{X}_D}{s_D/\sqrt{n}},$$

[1] `https://en.wikipedia.org/wiki/Student%27s_t-test#Dependent_t-test_for_paired_samples`

where $\overline{X}_D$ is the average of the differences between all pairs and s_D is the standard deviation of the differences. After the t value is computed, a p-value can be found in the table of values from the Student's t-distribution based on the estimated degrees of freedom (equal to $n - 1$).

If the reference and online computed datasets differed significantly, it would mean that the tool is not processing data the same way as when they are processed offline. Function `scipy.stats.ttest_rel`[2] was used for estimating the statistical significance. To test whether the results of our tool are significantly different from offline processing we chose the commonly used alpha level 5%. The function calculates *t-statistic* for two paired sets of data with the same number of samples together with the p-value. If the p-value is above 0.05, the two datasets are not differing with a statistical significance.

Question 2

> Is the tool robust enough for processing long-term signals?

The tool was tested on processing datasets from three patients flowing into it in a simulated real-time environment. Each of these datasets was several hours long, with the longest (patient 95) having about 13 hours. Such input signals may be noisy, contain extreme values or contain signals with dropped samples. Therefore, there was a risk of error occurrence during their processing. The tool allows for specifying sampling frequency and length of time windows in a number of samples passed into an Epycom function, which also determines an overall number of results in the database. Different values of these parameters were applied during the tests. We also tested processing large amounts of data when one of the datasets was sent repeatedly in a loop for several days.

Question 3

> Does the tool achieve similar performance as the referential 30 min. relaxed-state recording?

The long-term recordings are currently not being processed in FNUSA because of the technical demands. The iEEG epileptic biomarkers are derived from 30 min recordings when the patient is lying relaxed in bed. This approach has the advantage of eliminating movement artifacts, but such a strategy may not be sufficient since the latest research suggests the existence of cycles of epileptogenic activity in the brain [18]. There is a risk that the chosen time window for relaxed recording will fall into the period when the patient's brain shows only slight signs of epileptogenic activity. The assumption of epileptogenic cycles is further discussed and verified as a part of answering the questing 6.1.

After the processing of long recordings, results were pulled from the database. The following iEEG biomarkers were evaluated:

Relative entropy

The data initially stored in the database in 1 s time windows were divided into 30 min long segments (there were altogether 25 segments).

One segment thus consisted of 1800 windows. 152 pairs of channels were selected for the evaluation. The average REN was calculated for each of the pairs throughout all segments. That gave, in the end, 25x152 data rows.

[2] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_rel.html`

HFO

HFO waere detected on 10 s long time windows. Obtained data were again divided into 30 min segments. One time segment, therefore, consisted of 180 windows from the database. HFO occurrence is not distributed evenly among the channels. In theory, pathological channels should exert more HFO activity than non-pathological ones. Only HFO within frequency band $80 - 250\,Hz$ (also referred to as ripples) were chosen for the test because the lower frequencies are not considered epileptogenic and with higher frequencies, there is a risk of violation of the Nyquist theorem for the sampling frequency of 1000 Hz. 161 out of 172 channels were selected for evaluation. HFO were counted for each channel throughout all the segments.

Statistical evaluation

Channels P'6 and P'7 were labeled as pathological by the neurologist. The receiver operating characteristic (ROC) was computed for reference results, the as well as for individual segments. ROC curve illustrates the diagnostic ability of a binary classifier system at different threshold settings[3]. The graph is created by plotting the true positive rate (TPR) of the confusion matrix on the y-axis and the false positive rate (FPR) on the x-axis, computed as:

$$TPR = \frac{TruePositive}{TruePositive + FalseNegative}, \quad FPR = \frac{FalsePositive}{FalsePositive + TrueNegative},$$

TPR and FPR were computed by function
`sklearn.metrics.roc_curve`[4]. The function takes binary labels as the first argument and target scores as the second, returning FPR, TPR, and calculated thresholds. In our case, labels of individual channels (1 – pathological channel; 0 – non-pathological channel) were passed as the first argument, and the mean RENs, or the number of detected HFO in a given segment, were passed to the function as scores. Besides computing ROC for each time segment, an average ROC curve was computed from ROC curves of individual 30 min time segments.

The area under the curve (AUC) can be computed to represent ROC by a single number, which allows to statistically estimate the difference between the particular ROC curves. For calculating AUC, function `sklearn.metrics.auc`[5] was used. The function takes previously computed FPR and TPR and returns the area under the ROC curve. AUC of the ROC of each segment was computed, the as well as of the reference ROC. In the next two steps

(a) The mean ROC curve was statistically compared to the reference ROC curve

(b) Individual ROC curves were statistically compared to the reference ROC curve

Hanley-McNeil test [11] can estimate, whether the particular ROC curves are statistically different. It is computed as [10]:

$$z = \frac{A_1 - A_2}{\sqrt{SE_1^2 SE_2^2}},$$

[3] https://en.wikipedia.org/wiki/Receiver_operating_characteristic

[4] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html

[5] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.auc.html

where A_1 and A_2 represent values of AUC for the two compared ROC curves and SE_1 with SE_2 are the standard errors of the compared curves. The standard error of AUC is computed as:

$$SE(A) = \sqrt{\frac{A(1-A) + (n_P - 1)(Q_1 - A^2) + (n_N - 1)(Q_2 - A^2)}{n_P n_N}},$$

where n_P is number of samples (in our case channels) labeled as positive, n_N is number of samples labeled as negative, A represents AUC and

$$Q_1 = \frac{A}{2-A}; \quad Q_2 = \frac{2A^2}{1+A},$$

Question 4

> Does the ability to localize SOZ using biomarkers changes over time and can it outperform referential recording in some segments?

This question follows up the previous one, since the calculated statistics can tell if the results are varying throughout the time segments. To better visualize the differences between channels labeled as pathological and channels labeled as non-pathological, boxplot graph was used.

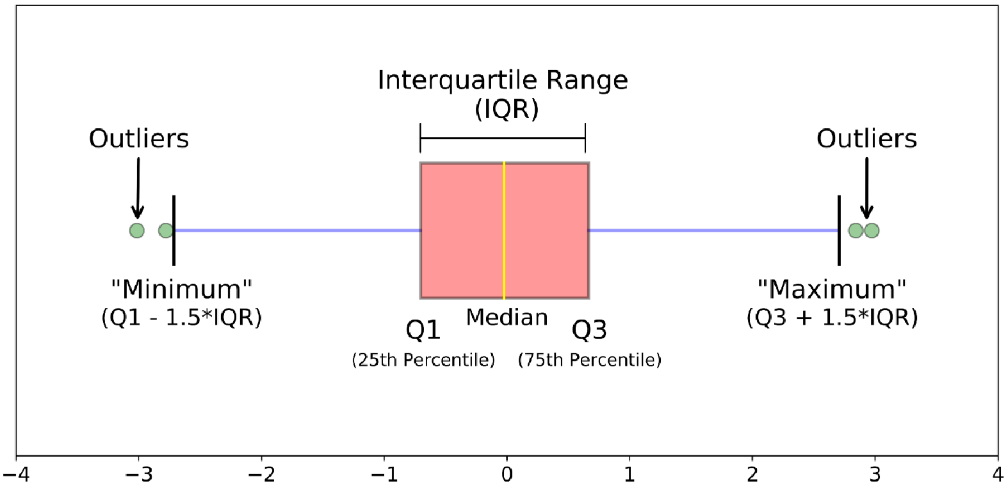


Figure 6.1: Boxplot [6]. The minimum is displayed in the graph as the first line of a left whisker. The left edge of the box is the first quartile. The right edge of the box is the third quartile. The line at the end of the right whisker is maximum.

A boxplot is a way of visualizing the distribution of the data within a five numbers summary[6]. The numbers are respectively: minimum (not the smallest number), first quartile – the middle number between the smallest number and the median, median, third quartile – the middle value between the median and the highest value and maximum (not the highest number). Figure 6.1 shows an illustration of the boxplot.

[6] https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51

6.2 Results

Question 1

Does our tool process the data same way as when they are processed offline?

The p-value of the difference between reference and online computed REN was 0.90, whereas the p-values of *mean signal power*, a *standard deviation of signal power*, and *maximum signal power* were all close to 1 by < 0.0001.

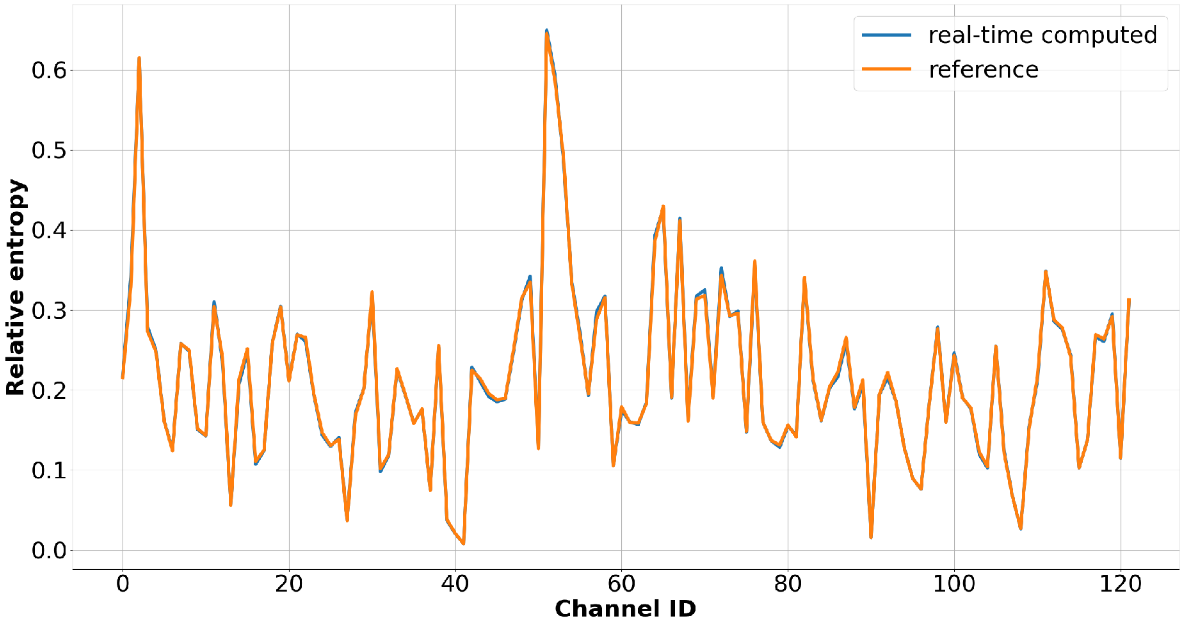


Figure 6.2: An average value of entropy per channel. Small differences between the datasets could be caused by possible packet loss or phase shift of incoming signal.

Question 2

Is the tool robust enough for processing long-term signals?

No crashes or other unexpected behavior was observed during the processing of long term signal. The tool performed well even after processing amounts of data for an extended period of several days.

Question 3

Does the tool achieve similar performance as the referential 30 min. relaxed-state recording?

Difference between mean ROC and refference ROC is not statistically significant in REN, with p-value = 0.2586. However, reference ROC was significantly bigger for HFO, with p-value < 0.001. ROC curves of individual segments were plotted in a single graph, together with ROC computed from the reference segment and mean ROC of all segments (figures 6.3 and 6.4).

Question 4

> Does the ability to localize SOZ using biomarkers changes over time and can it outperform referential recording in some segments?

Change of the AUC values in time is plotted in figures 6.5 and 6.6 together with one reference AUC. Differences between values of pathological and non-pathological channels, varying throughout the time, are displayed in figures 6.7 and 6.8. Table 6.1 shows p-values of statistical comparison of ROC curves throughout individual segments.

REN		
segment	**p-value**	**conclusion**
0	0.43022	equal
1	0.48739	equal
2	0.49045	equal
3	0.49837	equal
4	0.49286	equal
5	0.40636	equal
6	0.42012	equal
7	0.42012	equal
8	0.47334	equal
9	0.49045	equal
10	0.47909	equal
11	0.43768	equal
12	0.48480	equal
13	0.43154	equal
14	0.44376	equal
15	0.41908	equal
16	0.43768	equal
17	0.43768	equal
18	0.46167	equal
19	0.45576	equal
20	0.40636	equal
21	0.45576	equal
22	0.41521	equal
23	0.40070	equal
24	0.37719	equal

HFO		
segment	**p-value**	**conclusion**
0	<0.0001	ref. ROC greater
1	0.1739	equal
2	0.3330	equal
3	0.2063	equal
4	0.2772	equal
5	0.0009	ref. ROC greater
6	0.4394	equal
7	0.3524	equal
8	0.0002	ref. ROC greater
9	<0.0001	ref. ROC greater
10	<0.0001	ref. ROC greater
11	<0.0001	ref. ROC greater
12	0.0005	ref. ROC greater
13	0.0058	ref. ROC greater
14	<0.0001	ref. ROC greater
15	<0.0001	ref. ROC greater
16	<0.0001	ref. ROC greater
17	<0.0001	ref. ROC greater
18	0.0001	ref. ROC greater
19	<0.0001	ref. ROC greater
20	<0.0001	ref. ROC greater
21	<0.0001	ref. ROC greater
22	<0.0001	ref. ROC greater

Table 6.1: p-values of statistical comparison between ROC of real-time computed results and reference results for REN and HFO. Note, that there is less HFO segments than REN time segments. That is because the occurrence of HFO is irregular in time.

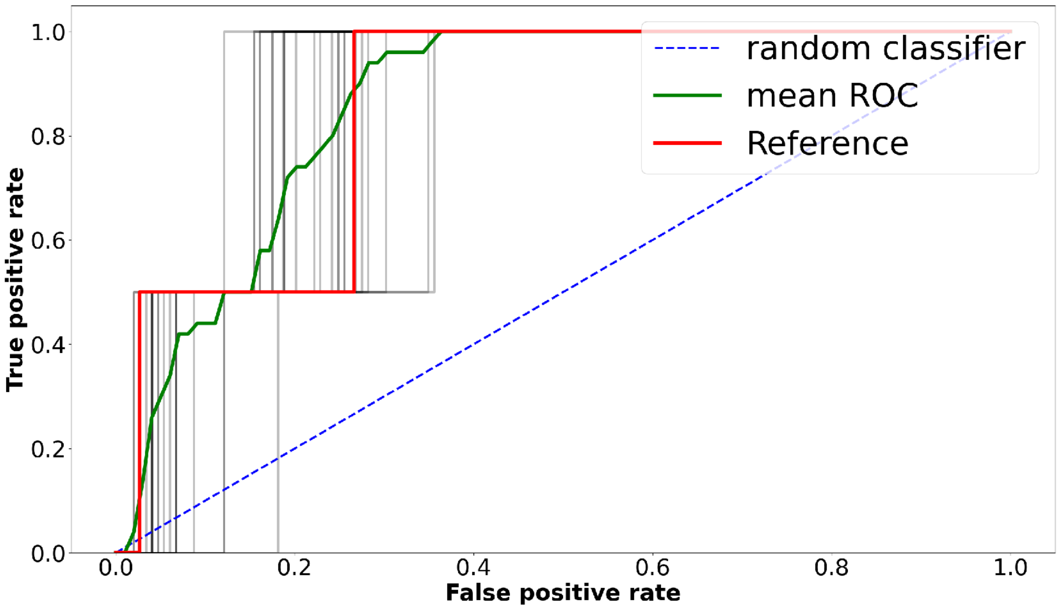


Figure 6.3: **Entropy** – ROC curves computed from the data pulled from the database in the pale color, together with ROC curve of offline computed data and mean ROC curve.

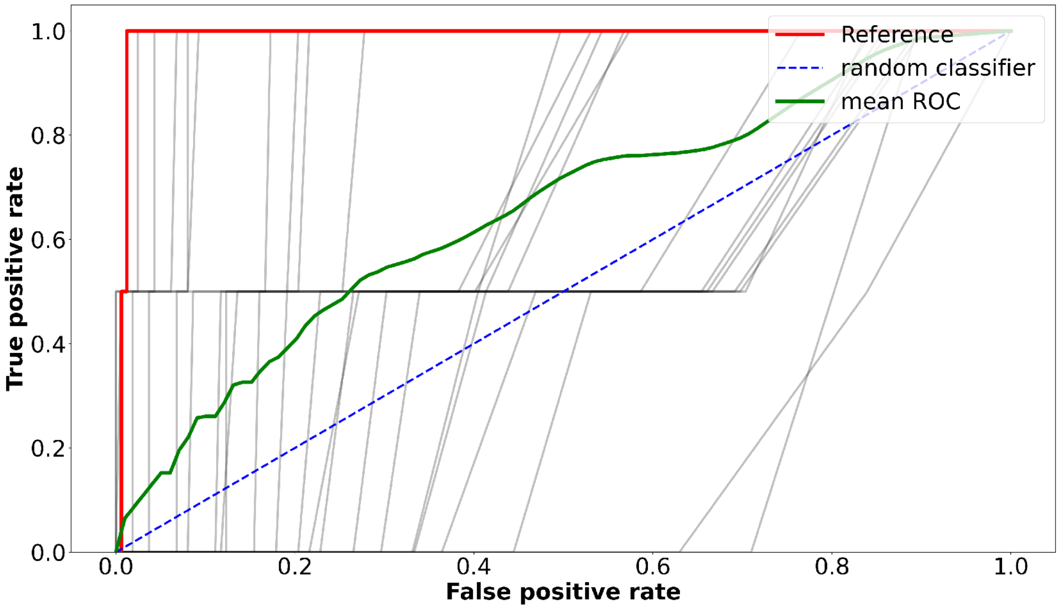


Figure 6.4: **HFO** – ROC curves compute from the data pulled from the database in the pale color, together with ROC curve of offline computed data and mean ROC curve.

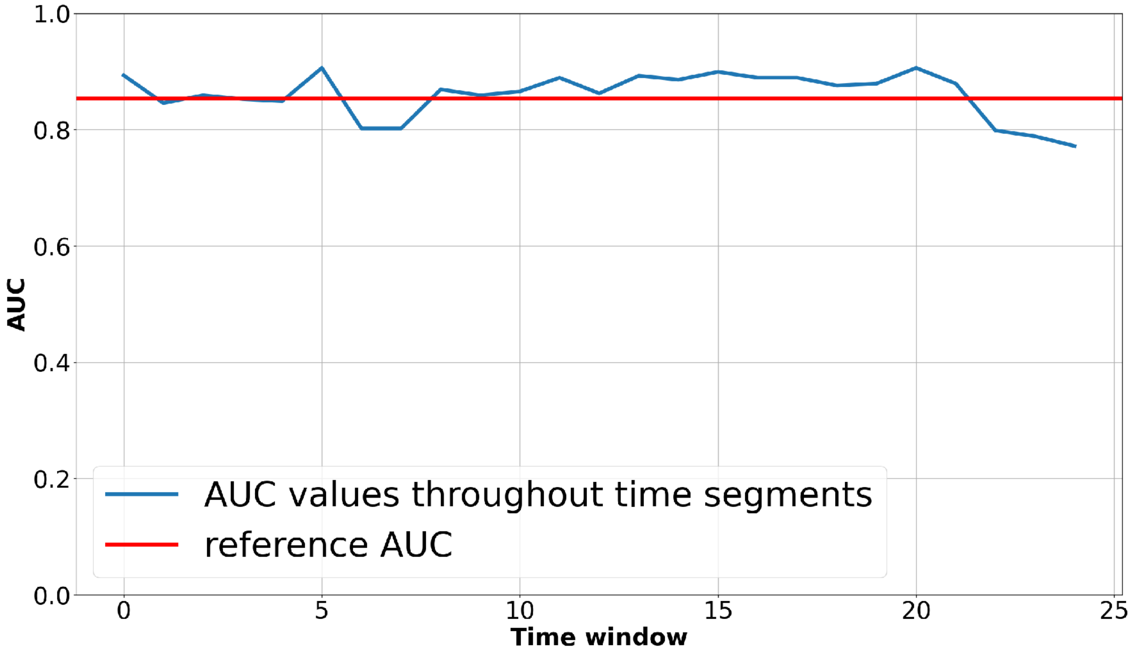


Figure 6.5: **Entropy** – AUC values computed from the data pulled from the database, together with one reference AUC of offline computed data.

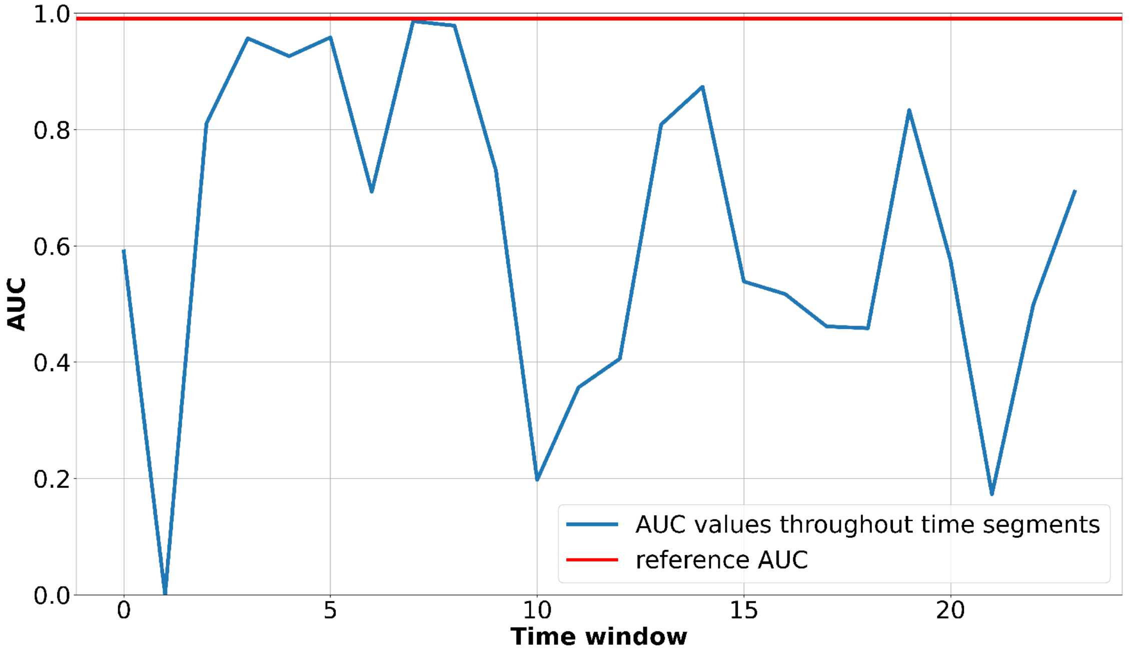


Figure 6.6: **HFO** – AUC values computed from the data pulled from the database, together with one reference AUC of offline computed data.

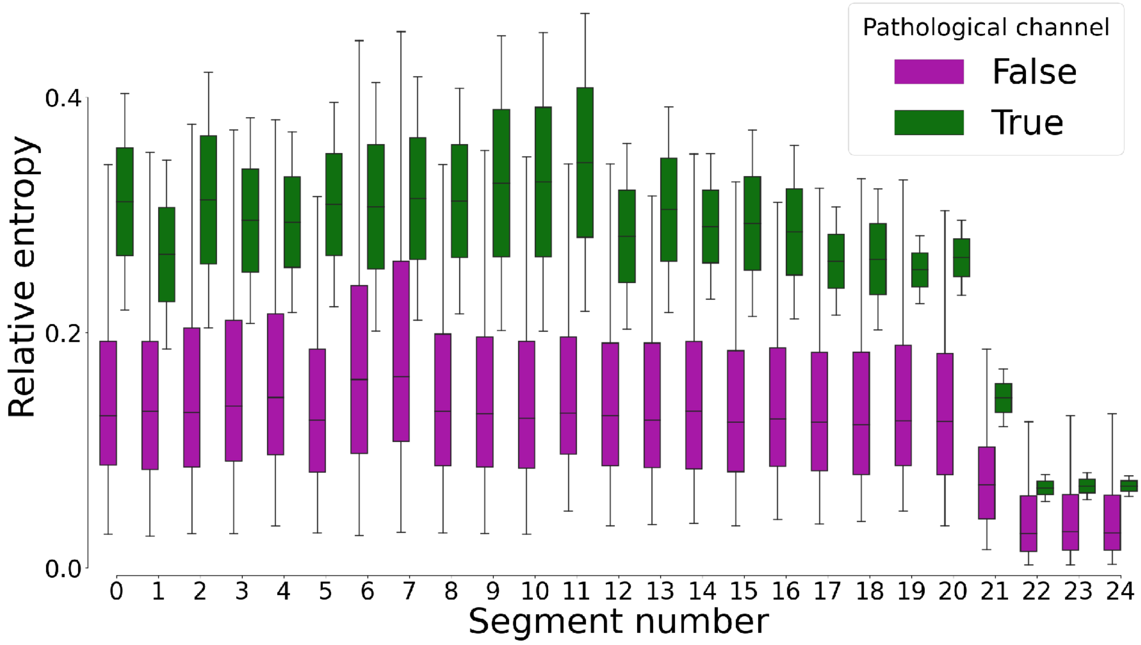


Figure 6.7: **Entropy** – Difference between values of relative entropy in pathological and non-pathological channels. Outlier points are not displayed on the graph.

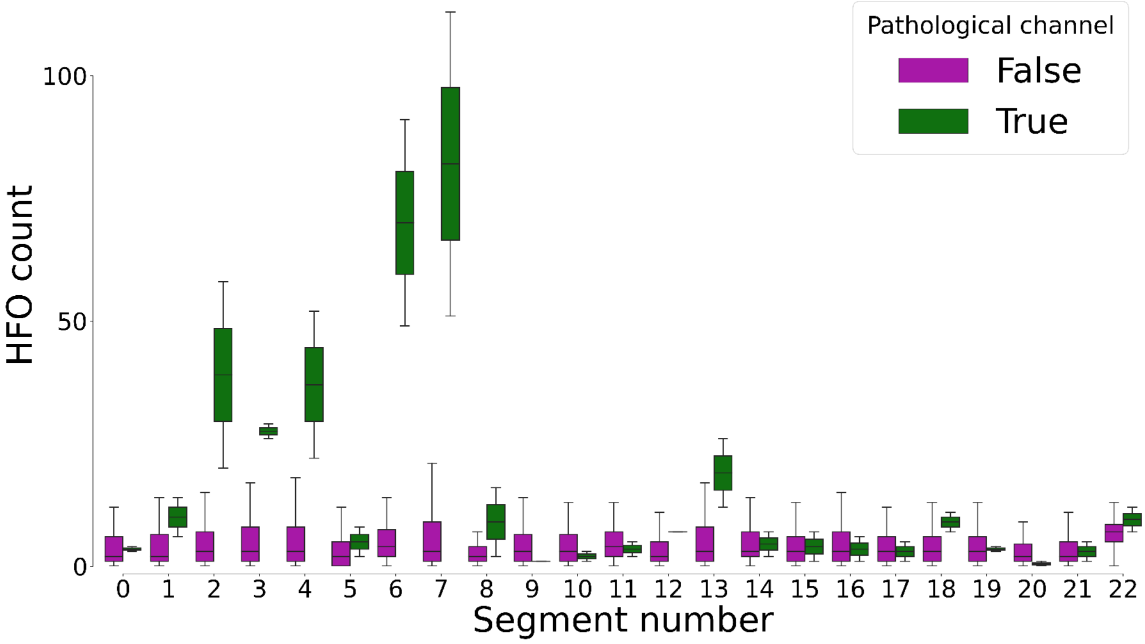


Figure 6.8: **HFO** – Difference between number of detected HFO in pathological and non-pathological channels. Outlier points are not displayed on the graph.

Chapter 7

Discussion

Answering the questions 1 and 2 from the previous Chapter was a straightforward task. We have proven that our pipeline processes signals the same way as when they are processed offline (question 1). Our tool is robust enough to be used in research and possibly in clinical practice (question 2). Answering question 3 was more difficult. While the mean ROC of real-time computed segments and ROC of reference results are statistically equal, which implies that our tool performs similarly in processing long recordings as offline processing of 30 min relaxed recordings, the mean ROC of HFO was statistically worse than the reference. It's important to point out, that long recordings are much noisier in comparison to the relaxed, short recordings since the patient is engaging in activities like eating, walking, speaking, etc. On the contrary, the patient is solely lying on a hospital bed during the making of the relaxed recording. With a closer look at the statistical differences of individual segments in table 6.1, we can see that HFO in segments 1, 2, 3, 4, 6, and 7 were, in fact, statistically equal to the reference results ($p > 0.05$). This finding offers an answer to the further question 4. Epileptogenic activity, but also artifacts and noise levels, seem to change in the brain over time. It can be observed further in the graphs, which are part of question 4. An estimation of the best moment for making a short relaxed recording based on a real-time analysis would be of great benefit in clinical practice.

7.1 Summary of work done

This thesis comes out of a need for the development of new tools in the field of pharmacoresistant epilepsy treatment. Patients suffering from this type of epilepsy do not respond to common anti-epileptic drugs, and in most cases, the only way to achieve a seizure-free state is resection of epileptogenic brain tissue. Precise localization of the epileptogenic zone (EZ) is therefore essential. As a part of this procedure, the patient must be implanted with intracranial EEG electrodes for up to four weeks, waiting for a seizure occurrence as the only way of localizing EZ up to this day. In some cases, even this process is not adequate to mark part of the brain for resection (this marked part is also called seizure onset zone, or SOZ) with enough confidence to perform the surgery. In such a case, presurgical assessment turns out to be wasteful. Therefore, a lot of effort is put into improving this process. The key parameters of this are time and accuracy. This thesis, made in cooperation with FNUSA-ICRC, lays a foundation for solving both mentioned aspects. We implemented a tool in Python that can process an incoming stream of EEG signals and extract desired EEG features from it in real-time. Since evaluating various EEG biomarkers turns out to bring better results in localizing SOZ [4], the precision of this task can be improved using our tool in combination with machine learning, or another decision-making algorithm. The real-time processing ensures that results are available immediately, and neurologists can make decisions right from the operating room during the recording. Unlike offline processing, during the online processing, data are stored in the database, well-structured for further evaluation. Our tool was tested on data from four patients. Three datasets were long recordings lasting for several hours, and one was a 30 min relaxed recording. To evaluate our tool, we asked four questions.

Calculated results were statistically compared and plotted in graphs. In the short recording, differences between offline and online computed data were minor (question 1). The tool processed all the datasets without crashes or other unexpected behavior (question 2). Functions for computing relative entropy and detecting HFO applied to the long datasets within our pipeline. To compare with offline computations, only 30 min offline computed relaxed recordings were available as a reference. The results were compared to the real-time processed data separated into 30 min long time segments. The values of REN after computing ROC were statistically equal to the reference results from 30 min relaxed recording. It implies that our tool is suitable for analyzing long-term recordings with similar performance as the offline analysis of short recordings (question 3). The ROC values of HFO were statistically worse than the reference ROC. That can be caused due to the lower quality of long-term recordings, which are subject to much bigger noise. The potential benefits of our tool still exceed its lower performance in HFO detection. That is also because of the findings after analyzing statistical differences in the individual segments. Despite the overall result, our tool performance was similar to the offline analysis in particular segments in HFO detection. This suggests the existence of time-varying epileptic cycles in the brain and offers a possibility of improved SOZ localization thanks to this knowledge (question 4).

Although there is a lot of ongoing research and development in the domains of EEGs and epilepsy around the world, not many institutions fully implemented online EEG processing yet. Our project can further provide a strong foundation for the further development of better localization and seizure prediction mechanisms at St Anne's Hospital Brno. Moreover, thanks to the immediate results, physicians can make decisions to ensure better comfort for the patient, right during the EEG capturing. The project can also help with the selection of the right time for the 30 min relaxed recording, which can facilitate further localization. Although the tool is fully functional in its current state, more work has to be put into its development in order to put it into practice.

7.2 Upcoming work

Throughout the development we made attempts to test the tool on a live patient simultaneously with standard recording, but we encountered technical problems with the local hospital network. Since the recording of EEG takes place on a different network than its processing within the FNUSA, and due to the strict policy towards the data flowing in and out, it is a problem to get permission for the transmission of the data from one network to another. This problem has to be solved to put the tool into practice. Furthermore, there is still work to be done to improve our pipeline and further develop its functions. The main points include:

- connecting the tool to the newly created database at BME FNUSA-ICRC. This goal is achievable in a matter of weeks after the thesis submission.
- to get the most of the potential from the tool, follow-up connection to machine learning algorithms will be ensured. That would allow real-time online localization.
- as described in section 4.4, we are facing the question of how to approach the signal filtering within the pipeline. The optimal and the most robust solution would be applying filters through a specific node, or nodes, as proposed in the original architectural design. That would require changes in the implementation of some Epycom functions which apply filters from within. Other functions, such as MVL, require signal filtered in different frequency bands and that also needs to be taken into account.
- there is also an incentive to create a web interface for visualizing the computed data from the database. The interface would include statistical graphs for easier assessment of the data by neurologists.
- tool could be further optimized to increase the speed of computing. Using optimizing tools, such as `Numba`[1] (just in time Python compiler) to improve the performance will be considered

7.3 Future prospects

Implementation of machine learning – online localization

With new research and advances in machine learning algorithms, it is possible that presurgical assessment will be fully automatized in the future. Our pipeline presents the perfect infrastructure for such a process. Machine learning-based algorithms, that can localize SOZ from interictal biomarkers already exist [4]. The time needed for signal capturing would be much shorter if there is no need to wait for the patient's seizure during his stay in the hospital. At the same time, machine learning algorithm could evaluate the data in real-time and immediate results from the localization would be available.

Implementation of seizure prediction algorithms

A bit different direction is the prediction of seizures, which is becoming another hot topic in epilepsy research. The mechanism for sending data in real-time and immediately processing them by a machine has been used already for a successful seizure prediction [7]. Our tool could be used for implementing such a seizure predicting device at FNUSA. This device would warn nurses of oncoming seizures and ensure enough time to prevent imminent danger to the patient.

Possible implementation on other institutions

Thanks to the close cooperation of FNUSA-ICRC with other institutes, our pipeline might be eventually used in one of them. This will likely happen at Mayo clinic[2]. Several studies from the field of epilepsy have been conducted in cooperation with these two institutes, such as [27] or [28].

[1] https://numba.pydata.org/

[2] https://www.mayoclinic.org/

Usage of the tool for neuroscience

Understanding the human brain is relevant not only for the treatment of epilepsy. This work may find use also in general research in the neuroscience field. One of the possible uses is implementation as a part of an instant neurofeedback device. Commercial non-invasive devices for neurofeedback are already available to the public[3] and with ongoing research in this area, the possible use of such devices will widen.

[3] `https://choosemuse.com/`

Bibliography

[1] Amiri, M., Frauscher, B. and Gotman, J. Phase-Amplitude coupling is elevated in deep sleep and in the onset zone of focal epileptic seizures. *Frontiers in Human Neuroscience*. jan 1 2016, vol. 10, no. 1. DOI: 10.3389/fnhum.2016.00387. ISSN 1662-5161.

[2] Ball, T., Kern, M., Mutschler, I., Aertsen, A. and Schulze Bonhage, A. Signal quality of simultaneously recorded invasive and non-invasive EEG. *NeuroImage*. 2009, vol. 46, no. 3, p. 708–716. DOI: https://doi.org/10.1016/j.neuroimage.2009.02.028. ISSN 1053-8119.

[3] Cimbálník, J., Hewitt, A., Worrell, G. and Stead, M. The CS algorithm: A novel method for high frequency oscillation detection in EEG. *Journal of Neuroscience Methods*. Elsevier BV. january 2018, vol. 293, no. 1, p. 6–16. DOI: 10.1016/j.jneumeth.2017.08.023.

[4] Cimbalnik, J., Klimes, P., Sladky, V. ladimir, Nejedly, P., Jurak, P. et al. Multi-feature localization of epileptic foci from interictal, intracranial EEG. *Clinical Neurophysiology*. 2019, vol. 130, no. 10, p. 1945–1953. DOI: https://doi.org/10.1016/j.clinph.2019.07.024. ISSN 1388-2457.

[5] Cimbalnik, J., Kucewicz, M. T. and Worrell, G. Interictal high-frequency oscillations in focal human epilepsy. *Current Opinion in Neurology*. Ovid Technologies (Wolters Kluwer Health). april 2016, vol. 29, no. 2, p. 175–181. DOI: https://doi.org/10.1097/wco.0000000000000302.

[6] Contributors to Wikimedia projects. *Electrocorticography*. Feb 18 2022. [Online; accessed 2022-04-15]. Available at: `https://en.wikipedia.org/wiki/Electrocorticography`.

[7] Cook, M. J., O'Brien, T. J., Berkovic, S. F., Murphy, M., Morokoff, A. et al. Prediction of seizure likelihood with a long-term, implanted seizure advisory system in patients with drug-resistant epilepsy: A first-in-man study. *The Lancet Neurology*. Elsevier BV. june 2013, vol. 12, no. 6, p. 563–571. DOI: 10.1016/s1474-4422(13)70075-9.

[8] Fisher, R. S., Acevedo, C., Arzimanoglou, A., Bogacz, A., Cross, J. H. et al. ILAE Official Report: A practical clinical definition of epilepsy. *Epilepsia*. 2014, vol. 55, no. 4, p. 475–482. DOI: https://doi.org/10.1111/epi.12550.

[9] Frauscher, B. Localizing the epileptogenic zone. *Current Opinion in Neurology*. Ovid Technologies (Wolters Kluwer Health). april 2020, vol. 33, no. 2, p. 198–206. DOI: 10.1097/wco.0000000000000790.

[10] Hajian Tilaki, K. O. and Hanley, J. A. Comparison of three methods for estimating the standard error of the area under the curve in ROC analysis of quantitative data. *Academic Radiology*. Elsevier BV. november 2002, vol. 9, no. 11, p. 1278–1285. DOI: 10.1016/s1076-6332(03)80561-5.

[11] Hanley, J. A. and McNeil, B. J. A method of comparing the areas under receiver operating characteristic curves derived from the same cases. *Radiology*. Radiological Society of North America (RSNA). september 1983, vol. 148, no. 3, p. 839–843. DOI: 10.1148/radiology.148.3.6878708.

[12] Herff, C., Krusienski, D. J. and Kubben, P. The potential of stereotactic-eeg for brain-computer interfaces: Current progress and future directions. *Frontiers in Neuroscience*. Frontiers Media SA. feb 27 2020, vol. 14, no. 1. DOI: https://doi.org/10.3389/fnins.2020.00123.

[13] Hülsemann, M. J., Naumann, E. and Rasch, B. Quantification of phase-amplitude coupling in neuronal oscillations: Comparison of phase-locking value, mean vector length, modulation index, and generalized-linear-modeling-cross-frequency-coupling. *Frontiers in Neuroscience*. jan 1 2019, vol. 0, no. 1. DOI: https://doi.org/10.3389/fnins.2019.00573.

[14] Ibrahim, G. M., Wong, S. M., Anderson, R. A., Singh Cadieux, G., Akiyama, T. et al. Dynamic modulation of epileptic high frequency oscillations by the phase of slower cortical rhythms. *Experimental Neurology*. Elsevier BV. january 2014, vol. 251, no. 1, p. 30–38. DOI: https://doi.org/10.1016/j.expneurol.2013.10.019.

[15] Janca, R., Jezdik, P., Cmejla, R., Tomasek, M., Worrell, G. A. et al. Detection of interictal epileptiform discharges using signal envelope distribution modelling: Application to epileptic and non-epileptic intracranial recordings - PubMed. *Brain topography*. jan 1 2015, vol. 28, no. 1. DOI: 10.1007/s10548-014-0379-1.

[16] Jiang, X., Bian, G.-B. and Tian, Z. Removal of artifacts from EEG signals: A review. *Sensors*. MDPI AG. feb 26 2019, vol. 19, no. 5, p. 987. DOI: 10.3390/s19050987.

[17] Joseph I. Sirven, S. C. S. *Electroencephalography (EEG)*. Aug 22 2013. [Online; accessed 2022-05-08]. Available at: `https://www.epilepsy.com/learn/diagnosis/eeg`.

[18] Karoly, P. J., Rao, V. R., Gregg, N. M., Worrell, G. A., Bernard, C. et al. Cycles in epilepsy. *Nature Reviews Neurology*. Springer Science and Business Media LLC. mar 15 2021, vol. 17, no. 5, p. 267–284. DOI: https://doi.org/10.1038/s41582-021-00464-1.

[19] Katzner, S., Nauhaus, I., Benucci, A., Bonin, V., Ringach, D. L. et al. Local origin of field potentials in visual cortex. *Neuron*. jan 15 2009, vol. 61, no. 1, p. 35–41. DOI: https://doi.org/10.1016/j.neuron.2008.11.016.

[20] Kucewicz, M. T., Cimbalnik, J., Matsumoto, J. Y., Brinkmann, B. H., Bower, M. R. et al. High frequency oscillations are associated with cognitive processing in human recognition memory. *Brain*. Oxford University Press (OUP). jun 11 2014, vol. 137, no. 8, p. 2231–2244. DOI: https://doi.org/10.1093/brain/awu149.

[21] Kuhlmann, L., Lehnertz, K., Richardson, M. P., Schelter, B. and Zaveri, H. P. Seizure prediction — ready for a new era. *Nature Reviews Neurology.* Springer Science and Business Media LLC. aug 21 2018, vol. 14, no. 10, p. 618–630. DOI: 10.1038/s41582-018-0055-2.

[22] Kumar, J. S. and Bhuvaneswari, P. Analysis of electroencephalography (EEG) signals and its categorization–a study. *Procedia Engineering.* Elsevier BV. 2012, vol. 38, no. 1, p. 2525–2536. DOI: https://doi.org/10.1016/j.proeng.2012.06.298. [Online; accessed 2022-03-19].

[23] Kumar, J. S. and Bhuvaneswari, P. Analysis of electroencephalography (EEG) signals and its categorization–a study. *Procedia Engineering.* Elsevier BV. 2012, vol. 38, no. 1, p. 2525–2536. DOI: https://doi.org/10.1016/j.proeng.2012.06.298.

[24] Libenson, M. H. *Practical approach to electroencephalography e-book.* Elsevier Health Sciences, mar 6 2012. 89–96 p. ISBN 0750674784.

[25] Malmivuo, J. and Plonsey, R. *Bioelectromagnetism. 13. Electroencephalography.* Jan 1 1995. [Online; accessed 2022-03]. Available at: `https://www.researchgate.net/publication/321094865_{}Bioelectromagnetism_{}13_{}Electroencephalography`.

[26] Nait Ali, A. *Advanced biosignal processing.* Springer Science & Business Media, apr 21 2009. 123–145 p. ISBN 354089506X.

[27] Nejedly, P., Cimbalnik, J., Klimes, P., Plesinger, F., Halamek, J. et al. Intracerebral EEG artifact identification using convolutional neural networks. *Neuroinformatics.* Springer Science and Business Media LLC. aug 13 2018, vol. 17, no. 2, p. 225–234. DOI: https://doi.org/10.1007/s12021-018-9397-6.

[28] Nejedly, P., Kremen, V., Sladky, V., Cimbalnik, J., Klimes, P. et al. Multicenter intracranial EEG dataset for classification of graphoelements and artifactual signals. *Scientific Data.* Springer Science and Business Media LLC. jun 16 2020, vol. 7, no. 1. DOI: https://doi.org/10.1038/s41597-020-0532-5.

[29] Parvizi, J. and Kastner, S. Promises and limitations of human intracranial electroencephalography. *Nature Neuroscience.* Springer Science and Business Media LLC. mar 5 2018, vol. 21, no. 4, p. 474–483. DOI: 10.1038/s41593-018-0108-2.

[30] Pohlmann Eden, B. and Weaver, D. F. The puzzle(s) of pharmacoresistant epilepsy. *Epilepsia.* 2013, vol. 54, s2, p. 1–4. DOI: https://doi.org/10.1111/epi.12174.

[31] Ryvlin, P., Cross, J. H. and Rheims, S. Epilepsy surgery in children and adults. *The Lancet Neurology.* Elsevier BV. november 2014, vol. 13, no. 11, p. 1114–1126. DOI: 10.1016/s1474-4422(14)70156-5.

[32] Sivakumar, S. *The Wave - The characteristics of an EEG — Firstclass.* Aug 9 2017. [Online; accessed 2022-03-19]. Available at: `https://www.firstclassmed.com/articles/2017/eeg-waves`.

[33] Uher, D., Klimes, P., Cimbalnik, J., Roman, R., Pail, M. et al. Stereo-electroencephalography (SEEG) reference based on low-variance signals. In: IEEE. *2020 42nd Annual International Conference of the IEEE Engineering in*

Medicine &amp; Biology Society (EMBC). July 2020. DOI: 10.1109/EMBC44109.2020.9175734.

[34] Zuo, R., Wei, J., Li, X., Li, C., Zhao, C. et al. Automated detection of high-frequency oscillations in epilepsy based on a convolutional neural network. *Frontiers in Computational Neuroscience*. jan 1 2019, vol. 13, no. 1. DOI: https://doi.org/10.3389/fncom.2019.00006. ISSN 1662-5188. Available at: `https://www.frontiersin.org/articles/10.3389/fncom.2019.00006/full`.

Appendix A

Results from the ramaining patients

Patient 83

Channels L1 and L2 were labeled by neurologist as pathological. Refference ROC of REN was significantly greater than mean ROC of real-time computed results with p-value= 0.004. The mean ROC of HFO was significantly greater than reference ROC (p-value< 0.0001) which indicates a better performance of our tool in this patients HFO evaluation. Results for individual segments are available in table A.1.

REN		
segment	**p-value**	**conclusion**
0	0.48766	equal
1	0.33714	equal
2	0.37527	equal
3	0.38222	equal
4	0.43754	equal
5	0.40435	equal
6	0.44667	equal
7	0.37527	equal
8	0.30378	equal
9	0.29859	equal
10	0.26467	equal
11	0.44557	equal

HFO		
segment	**p-value**	**conclusion**
0	<0.0001	our ROC greater
1	0.35350	equal
2	0.13217	equal
3	0.17359	equal
4	0.01503	our ROC greater
5	0.09246	equal
6	0.02025	our ROC greater
7	0.00082	our ROC greater
8	0.09655	equal
9	0.03009	our ROC greater
10	0.0573	equal

Table A.1: p-values of statistical comparison between ROC of real-time computed results and reference results for REN and HFO. Note, that there is less HFO segments than REN time segments. That is because the occurrence of HFO is irregular in time.

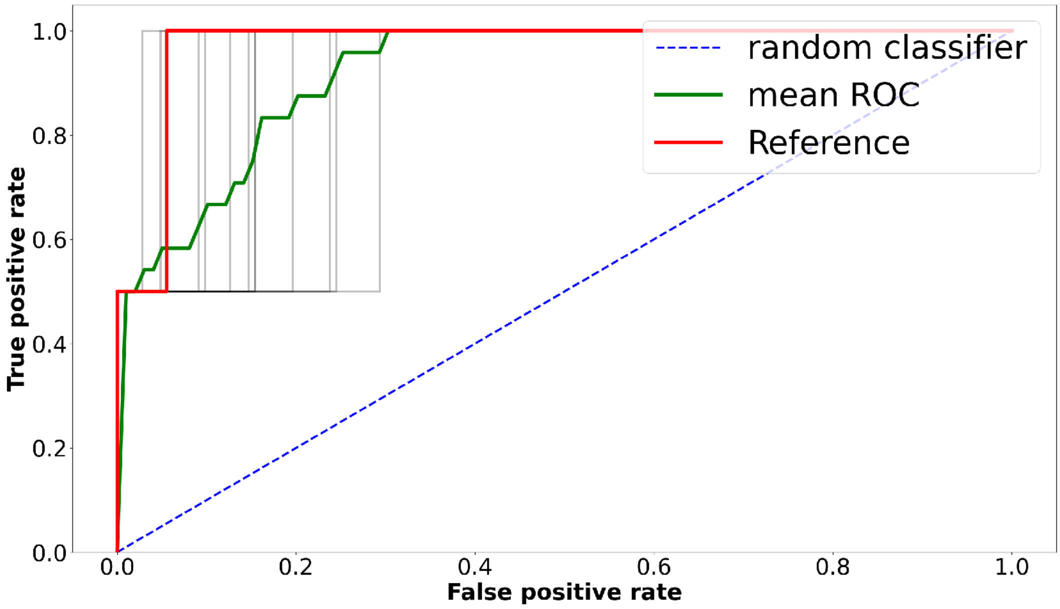


Figure A.1: **Patient 83 entropy** – ROC curves computed from the data pulled from the database in the pale color, together with ROC curve of offline computed data and mean ROC curve.

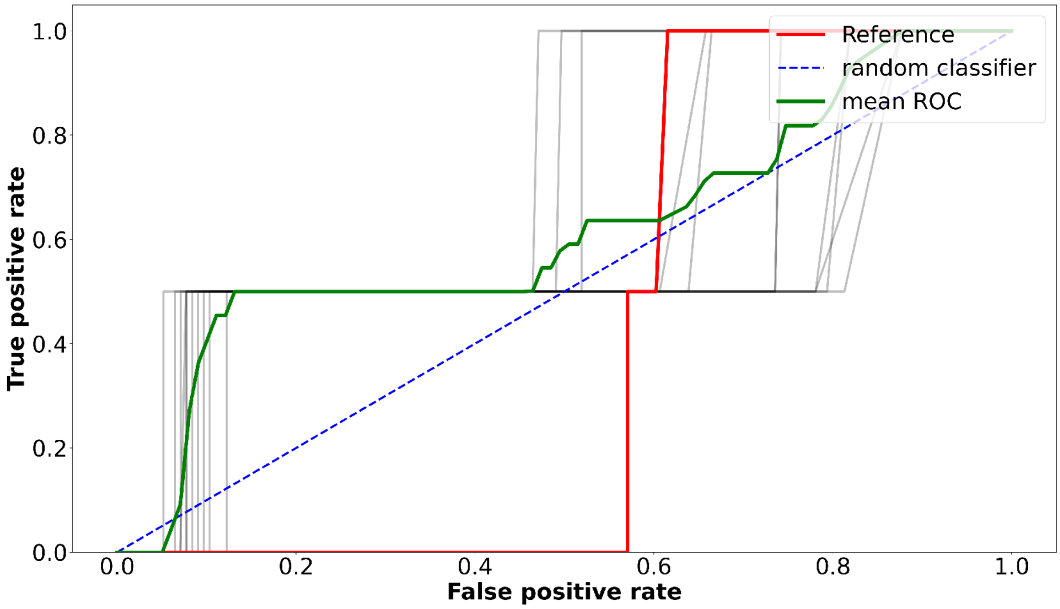


Figure A.2: **Patient 83 HFO** – ROC curves compute from the data pulled from the database in the pale color, together with ROC curve of offline computed data and mean ROC curve.

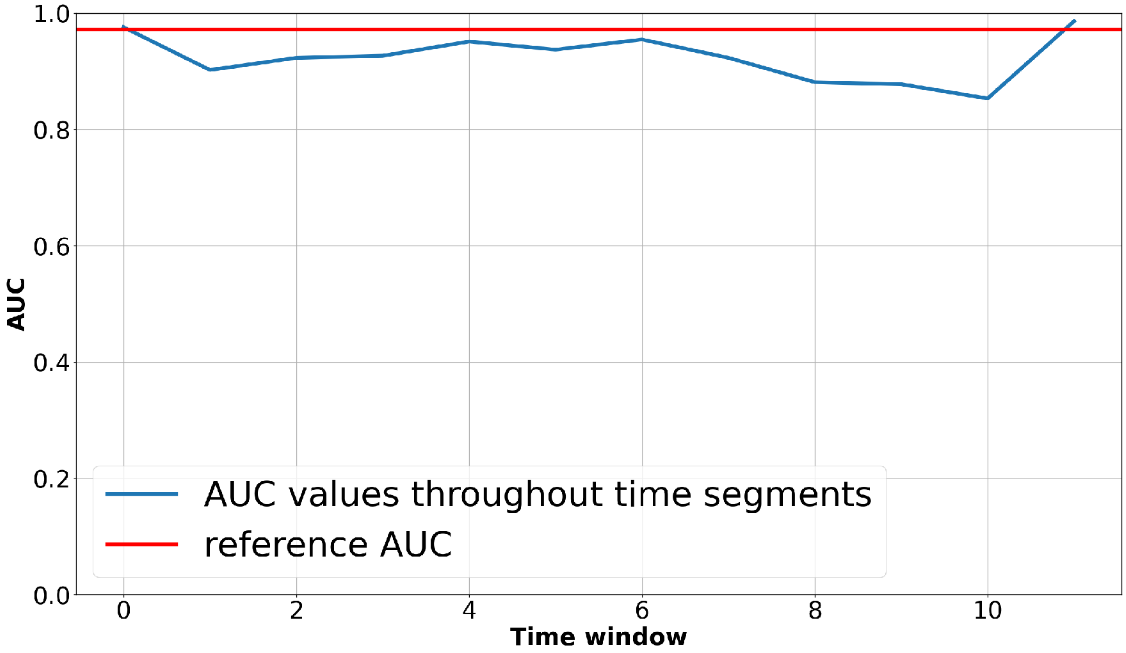


Figure A.3: **Patient 83 entropy** – AUC values computed from the data pulled from the database, together with one reference AUC of offline computed data.

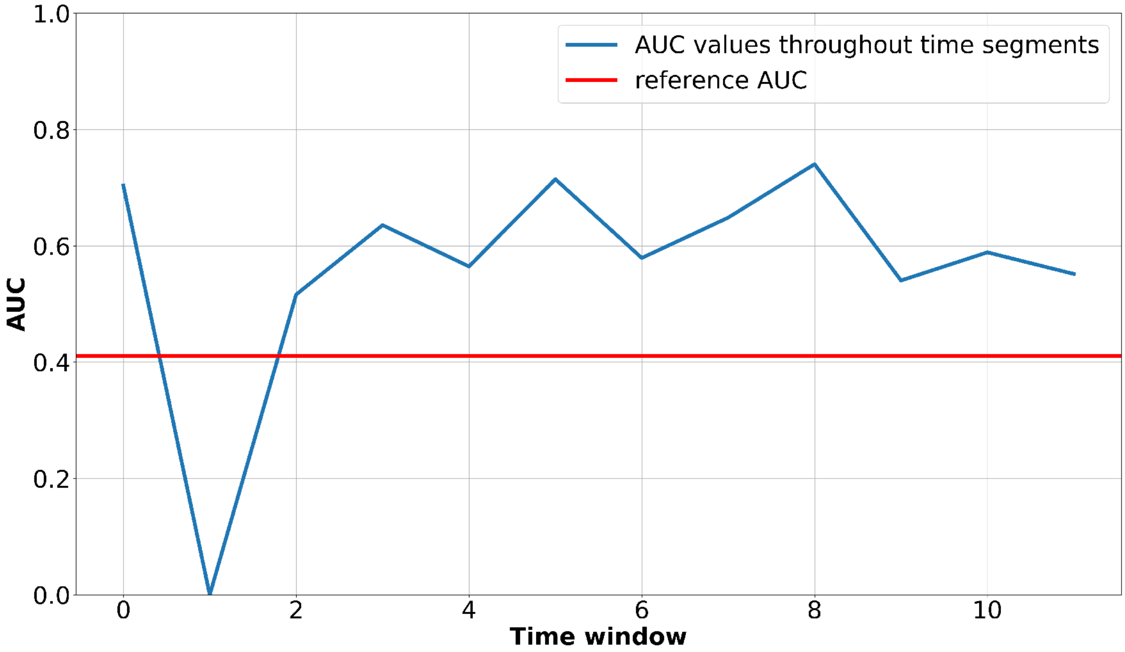


Figure A.4: **Patient 83 HFO** – AUC values computed from the data pulled from the database, together with one reference AUC of offline computed data.

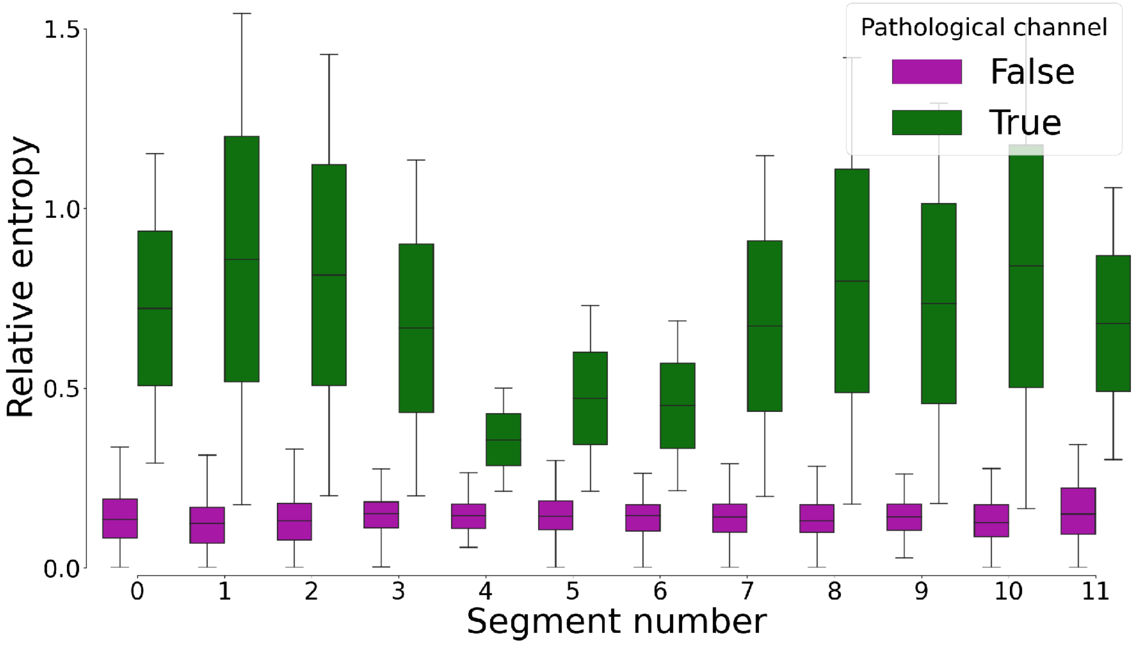


Figure A.5: **Patient 83 entropy** – Difference between values of relative entropy in pathological and non-pathological channels. Outlier points are not displayed on the graph.

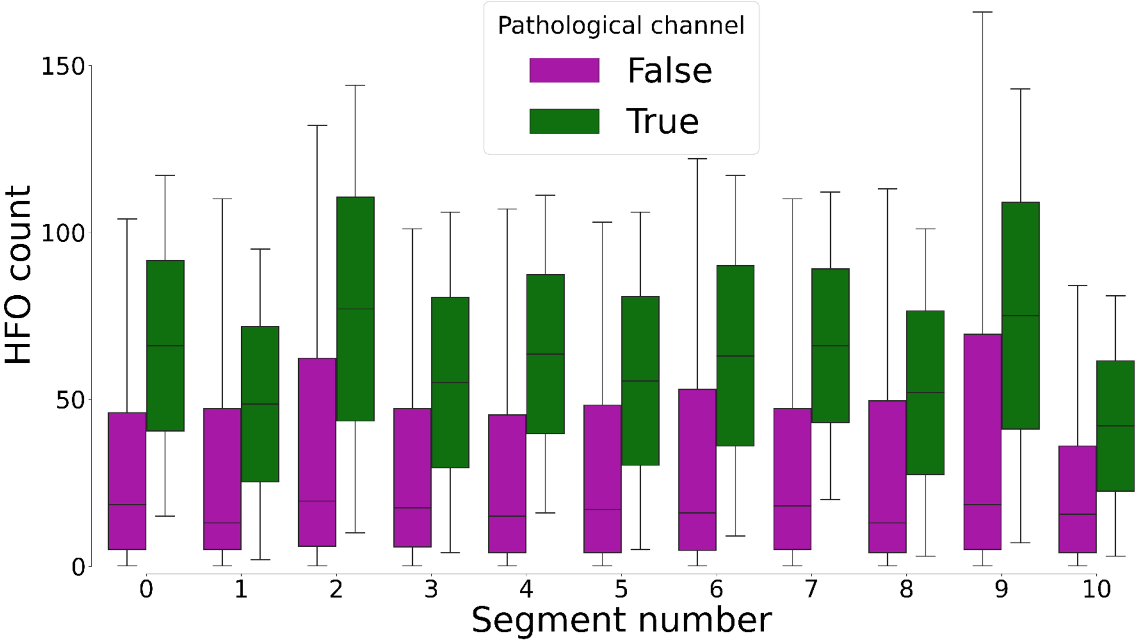


Figure A.6: **Patient 83 HFO** – Difference between number of detected HFO in pathological and non-pathological channels. Outlier points are not displayed on the graph.

Patient 79

Channels B'1, B'2 and B'3 were labeled by neurologist as pathological. Refference ROC of REN was significantly greater than mean ROC of real-time computed results with p-value$<$ 0.0001. Refference ROC was also significantly greater than mean ROC in HFO (p-value$<$ 0.0052). Results for individual segments are available in table A.2.

REN		
segment	**p-value**	**conclusion**
0	0.06957	equal
1	0.09702	equal
2	0.19861	equal
3	0.04517	ref. ROC greater
4	0.00350	ref. ROC greater
5	0.28193	equal
6	0.36685	equal
7	0.38582	equal
8	0.44736	equal

HFO		
segment	**p-value**	**conclusion**
0	<0.0001	ref. ROC greater
1	0.37540	equal
2	0.42677	equal
3	0.28971	equal
4	0.03874	ref. ROC greater
5	0.07084	equal
6	<0.0001	ref. ROC greater
7	<0.0001	ref. ROC greater

Table A.2: p-values of statistical comparison between ROC of real-time computed results and reference results for REN and HFO. Note, that there is less HFO segments than REN time segments. That is because the occurrence of HFO is irregular in time.

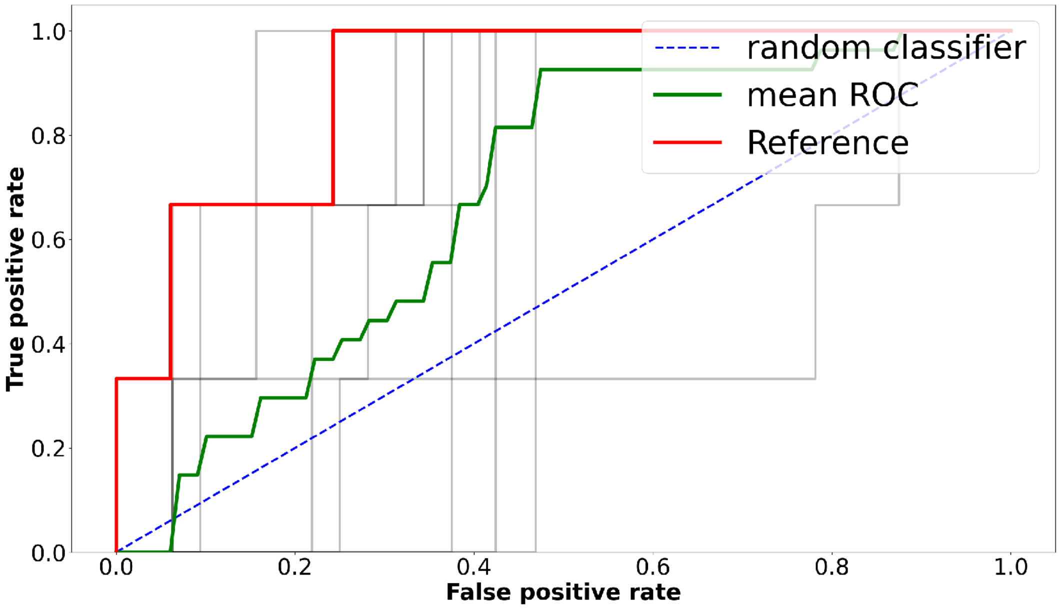


Figure A.7: **Patient 79 entropy** – ROC curves computed from the data pulled from the database in the pale color, together with ROC curve of offline computed data and mean ROC curve.

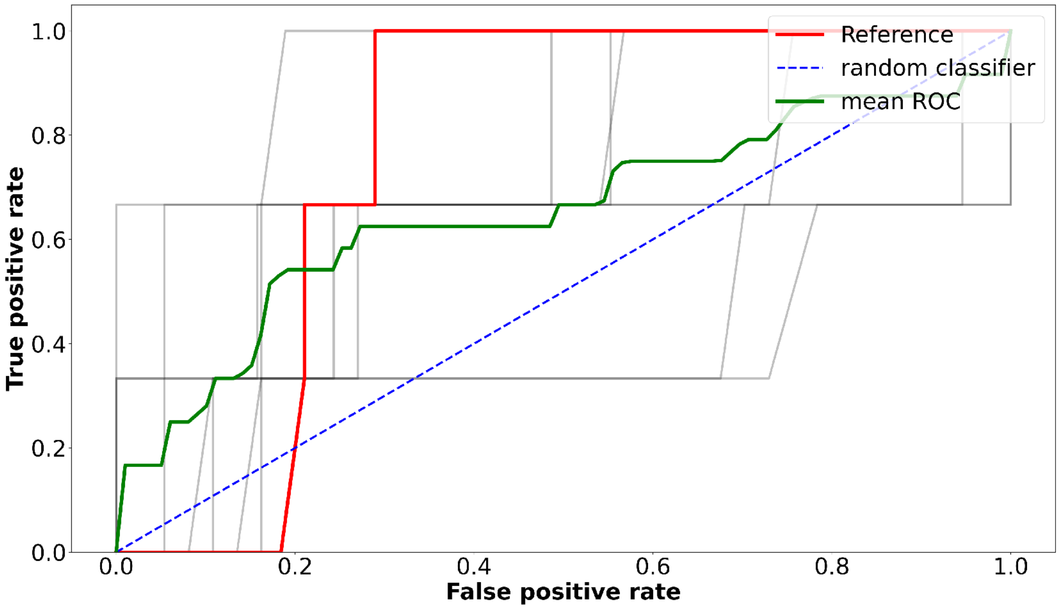


Figure A.8: **Patient 79 HFO** – ROC curves compute from the data pulled from the database in the pale color, together with ROC curve of offline computed data and mean ROC curve.

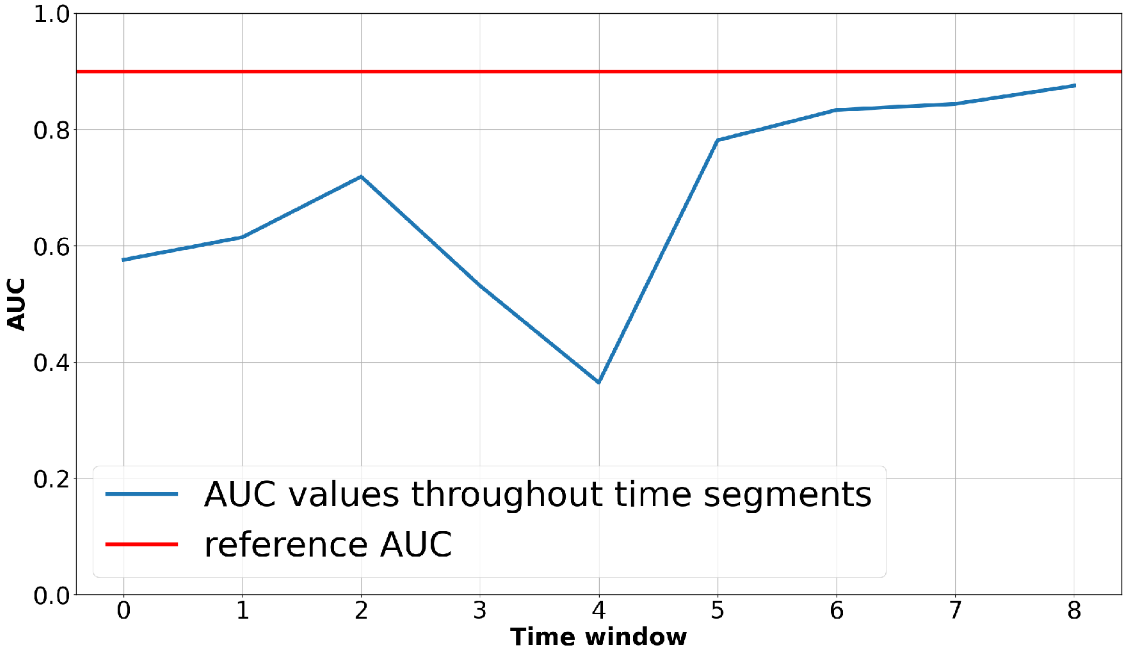


Figure A.9: **Patient 79 entropy** – AUC values computed from the data pulled from the database, together with one reference AUC of offline computed data.

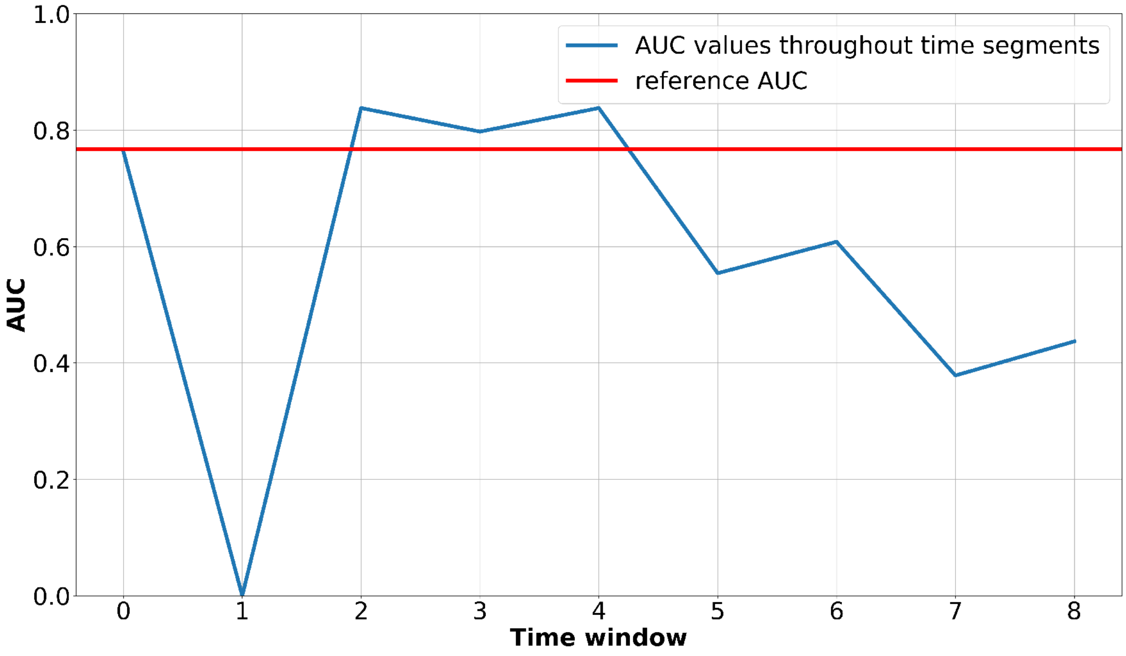


Figure A.10: **Patient 79 HFO** – AUC values computed from the data pulled from the database, together with one reference AUC of offline computed data.

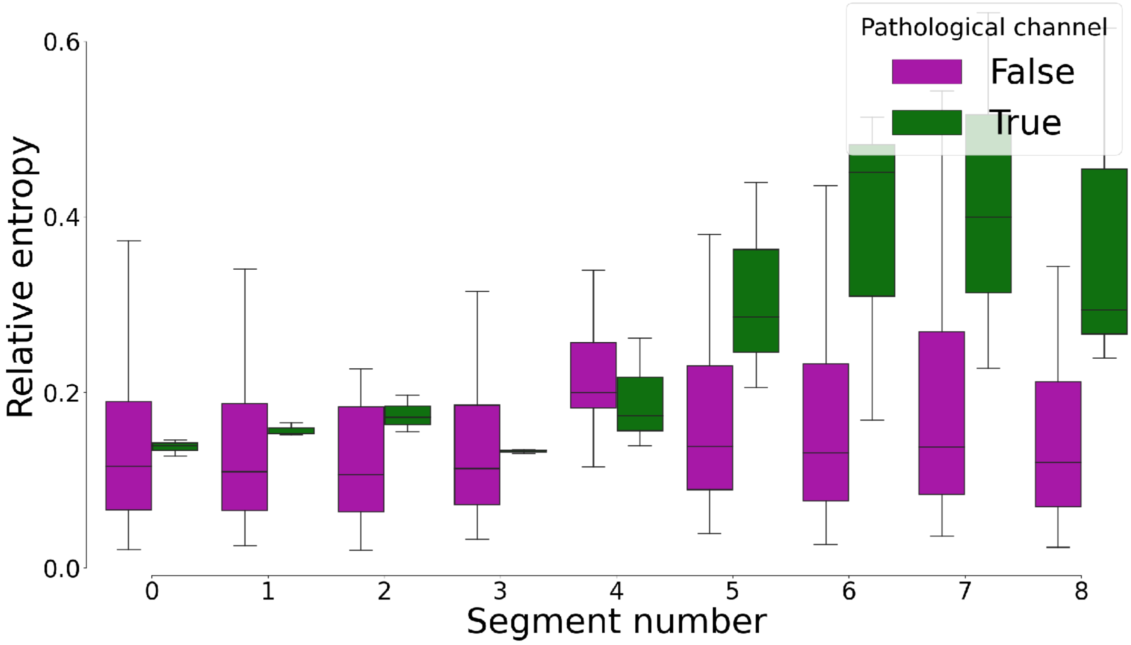


Figure A.11: **Patient 79 entropy** – Difference between values of relative entropy in pathological and non-pathological channels. Outlier points are not displayed on the graph.

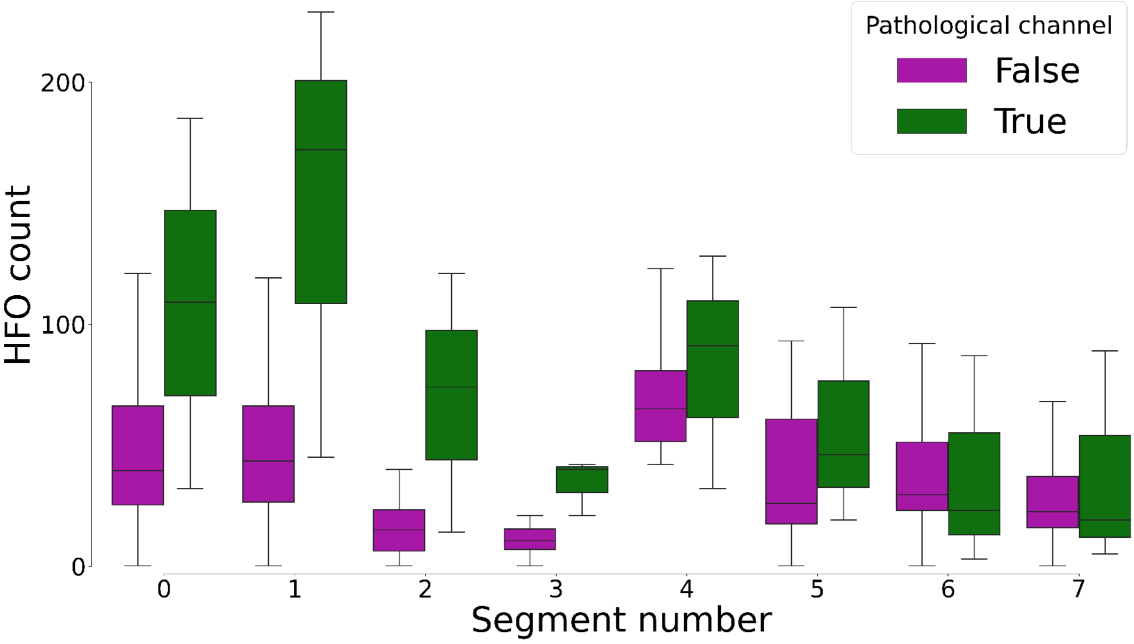


Figure A.12: **Patient 79 HFO** – Difference between number of detected HFO in pathological and non-pathological channels. Outlier points are not displayed on the graph.

Appendix B

Contents of SD card

- `\mepior`:
  - ∗ `\mepior` – mepior library
  - ∗ `pipeline_single.py` – single-process pipeline
  - ∗ `pipeline_manager.py` – distributed pipeline
- `\valuation`:
  - ∗ `valuate.py` – script used for evaluation of question 1 from Chapter 6
  - ∗ `compute_statistics.py` – script used for evaluation of questions 2 and 3 from Chapter 6
  - ∗ `get_channel_names.py`
- `\eeds` – simulation of real-time data flow:
  - ∗ `\build`
  - ∗ `\src`
  - ∗ `\work` – contains configurations files and usage information
- README.md – contains also installation guide