Università di Padova

Depratment of Mathematics "Tullio Levi-Civita"

Master Degree in Computer Science

Forecasting epileptic seizures from electroencephalograms using Deep Neural Networks

Master Degree

Supervisor

Prof.Alberto Testolin

Student

Marco Pozza

Academic year 2023-2024

Credi in quel che ami, continua a farlo,
e ti porterà dove hai bisogno di andare.
Natalie Goldberg

A mia nonna Rosy

Declaration

I, Marco Pozza, hereby declare that this thesis titled "Forecasting epileptic seizures from electroencephalograms using Deep Neural Networks", is my own work, conducted under the supervision of professor Testolin Alberto at University of Padova.

I confirm that this work is original and has not been submitted in any form for another degree or diploma at any university or other institution of higher education.

I acknowledge the assistance and contributions of Professor Alberto Testolin and Dr. Sina Shafiezadeh in the completion of this research.

All sources of information used in this thesis have been appropriately acknowledged and cited.

Padova, 15 December 2023

Marco Pozza

Abstract

The occurrence of epileptic seizures is a problem that makes everyday life very difficult for people who suffer from this disorder. This is especially true for patients whose disorder cannot be cured with therapies or drugs. In this work, I studied and developed different types of recurrent neural network (RNNs) that, through the analysis of electroencephalography (EEG) signals, aimed to correctly predict the onset of a seizure by detecting the preictal brain state and differentiating it from the interictal state within minutes before the actual epileptic attacks. The two proposed models use different approaches to extract relevant temporal features from raw EEG signals, which are then used for prediction purposes. Many different kinds of experiment and evaluation methods were used to assess the capacity of the model to correctly predict the seizures' onsets, leading to promising results that can be further enhanced with more sophisticated models that could extract more relevant temporal features.

Acknowledgements

Un sentito grazie al mio relatore, Alberto Testolin, per la sua guida preziosa e la sua disponibilità. Per avermi fornito tutti i materiali necessari alla stesura della tesi e per avermi accompagnato in questo percorso con competenza e passione.

Desidero esprimere la mia gratitudine al Dr. Sina Shafiezadeh, con il quale ho avuto il piacere di collaborare. I suoi preziosi consigli e il suo prezioso supporto mi sono stati di grande aiuto per portare avanti questo studio, nel quale ero totalmente un neofita.

Desidero ringraziare la mia famiglia per avermi sostenuto negli anni dei miei studi, per avermi spronato a impegnarmi e migliorare facendomi sempre credere nei miei sogni anche se, a volte, sembravano ridicoli.

Voglio ringraziare di cuore la mia ragazza Genny che, da quando l'ho conosciuta, non ha mai smesso di supportare il mio lavoro e sopportare i miei momenti di crisi spronandomi a non arrendermi di fronte alle difficoltà. La ringrazio soprattutto per avermi ascoltato senza giudicare e dato consigli nel modo più sincero e amorevole possibile.

Infine ringrazio i miei amici che anche se non in modo diretto, mi hanno accompagnato duranti tutti gli anni di studio. Grazie per essere sempre stati al mio fianco e per avermi fatto sorridere e affrontare con più leggerezza questo percorso.

Padova, 15 December 2023 Marco Pozza

Contents

List of figures

List of tables

1 | Introduction

The brain generates electrical activity due to communication between cells in the nervous system. Unfortunately, this electrical activity can be characterized by an abnormal release of energy leading to episodes of involuntary movement that can affect a single part of the body (partial) or the entire body (generalized). In more extreme cases, it can even cause a loss of consciousness. This disease is called Epilepsy. Epilepsy is one of the most common neurological diseases that affects approximately 50 million people worldwide [17] and between 1990 and 2015 we had an increase of 11.6% in deaths caused by it (from 112,000 to 125,000 [26]).

Patients spend most of their lives uncertain when and where epilepsy will occur, and one of the main problems is uncontrollability, which can cause injuries, social disability, and mortality, affecting their quality of life. Epilepsy can be controlled with antiepileptic drugs, but 1 in 3 patients have drug-resistant epilepsy [22], so alternative treatment is required. Analysis of electroencephalograms (EEG) shows that specific alterations in brain signals can be observed before epileptic onset, giving the possibility of creating methods based on artificial intelligence that can detect and anticipate seizure using EEG signals as input [1]. The main idea of these methods is based on the fact that AI methods can discover patterns in brain activity registered through EEGs and discriminate between normal and abnormal brain activity. The ability to predict in advance the onset of a seizure could be a huge improvement in the daily lives of these patients because they can be aware of the epileptic attack and take the right preventive treatments, such as electrical stimulation or drug delivery. Even if these precautions do not work, at least patients would be put in safety by healthcare professionals. The task in which we are interested is called seizure prediction, which differs from seizure detection. *Seizure detection* is a binary classification problem that aims to distinguish between abnormal brain activity and normal brain activity. In this kind of task, there are no predictive aspects because we already know which abnormalities are present in the EEG signals of a patient. *Seizure prediction* instead is a much more challenging problem because we need to distinguish normal brain activity from the one that precedes, within a certain range of seconds or minutes, an actual seizure. The wider the range, the greater the forecast, but the task will be more complicated.

The core of this master's thesis is on the latter task, using recurrent neural networks (RNNs), a type of computational model that excels at processing sequential data to

extract temporal features and patterns. The research presented here focuses on:

- the development of automated procedures that allow the processing of the raw EEG signals from each individual patient. This can be used to generate different types of datasets, organized by patient, for each specific experiment that I need to perform. The data were preprocessed in order to be compatible with other experiments performed by my supervisor and his collaborators;
- a comprehensive study on various types of RNNs, with particular emphasis on long short-term mermory (LSTM) and gated recurrent unit (GRUs). Subsequently, I used this knowledge to develop two distinct models. One model exclusively uses LSTM, while the other is a combination of convolutional neural network layers (CNN) and GRUs. These models were designed to autonomously extract meaningful temporal features, eliminating the necessity for manual feature extraction from the raw EEG data;
- the study of different feature selection procedures to identify the most suitable one for EEG signal processing, allowing me to exclude less relevant channels and retain only the most informative ones. This procedure is then employed to generate new datasets for conducting other experiments;
- exploring the efficacy of the two proposed models. To this end, I conducted a series of experiments using datasets created by manipulating the size of the signal window, varying degrees of overlap between the EEG segments, and different down-sampling frequencies. For each of these datasets, I have performed experiments that involve the use of all patients and the evaluation of the models trained with all the data but one left out patient.

This document is organized as follows.

Chapter 2: Epileptic Disorder. This chapter focuses on epileptic disorders from a medical point of view. This paper will begin by providing an overview of the disease, including its symptoms, how it affects the daily life of those affected, and how it can be treated. We will then discuss why a seizure prediction system is necessary and in what situations it is beneficial. Finally, we will emphasize the importance of such a system.

Chapter 3: Neural Networks Building Blocks. This chapter focuses on recurrent neural networks. Firstly, they are introduced by explaining the mathematical principles, how they work, and why they are preferred in contexts where sequential data is used. Following this generic background, the chapter discusses LSTMs and GRUs, what they are, how they work, and what sets them apart from conventional RNNs.

Chapter 4: Problem definition and experimental setup. This chapter focuses on the data used throughout the research and experiments. First, it introduces what EEGs are and how they can be used with deep learning models. Then, there is a brief overview of the preprocessing steps and methods applied to

the data to make it compatible with RNN models. Finally, we present all the experiments conducted and discuss their significance for the seizure prediction task.

Chapter 5: Models and Optimization. This chapter focuses on the models utilized in the experiments. Both models are introduced, and their strengths and weaknesses are discussed in the context of the seizure prediction task, particularly when dealing with large datasets.

Chapter 6: Results. In this chapter, the results of the experiments, divided by dataset, are presented and discussed, highlighting the prediction capabilities of the models along with their limits.

Chapter 7: Conclusions. This final chapter provides a comprehensive summary of the document, highlighting the findings derived from the extensive experimental work. These insights form the basis for concluding remarks regarding the predictive potential of RNNs in delicate contexts like neurological diseases, and offer broader conclusions about the overall research.

2 | Epileptic Disorder

In this chapter, I introduce epilepsy as a neurological disorder, focusing on the clinical aspects. We discuss what epilepsy is, how seizures start, the different types of epilepsy, and their consequences on the daily lives of affected patients. I also introduce possible treatments, why some of them may not relieve seizures and then the need for a procedure to forecast seizure onset, which would allow healthcare providers to prepare patients and themselves for impending seizures.

2.1 Overview

Epilepsy is a neurological disorder characterized by susceptibility to seizures. It is one of the most common brain disorders that affects 1% of the world's population [17].
It is characterized by recurrent seizures, which are abnormal electrical discharges at the level of the cerebral cortex. Seizures can be asymptomatic or cause episodes of involuntary movement that may involve a part of the body (partial) or the entire body (generalized). Seizures can also be accompanied by loss of consciousness and control of bowel or bladder function[4].
To be considered epilepsy, two or more seizures must occur at least 24 hours apart and must be spontaneous (not triggered by a known factor). In many cases, epilepsy is caused by an unknown condition (idiopathic), but other brain disorders (such as tumors or malformations) can cause symptomatic epilepsy[12].
The recurrent and unpredictable nature of seizures makes it difficult for people with epilepsy to live normal lives. They may have a seizure at any time, which can cause falls, injuries, and even death for themselves or others, but pharmacologic or surgical therapy can control seizures in 80% of cases.

2.2 Classification and Symptoms

Epileptic seizures are classified according to three different characteristics [21]:

1. origin of the seizure in the brain;

2. degree of awareness during seizure;

3. level of body movement.

On the basis of the first characteristic, a seizure could be focal or generalized. A **focal seizure** originates from a specific region of the cerebral cortex that has abnormal excitability and the manifestation depends on the affected area. For example, if the affected area deals with language, the patient will have difficulties speaking and/or understanding speech. Instead, a **generalized seizure** occurs in both hemispheres of the brain at the same time. A focal seizure can evolve into a generalized one.
The second feature implies the degree of awareness, the consciousness of the self and the surroundings; meanwhile, the last feature implies that a seizure could imply the movement of one or more parts of the body (motor) or not (nonmotor).
There is one last type: **seizures of unknown onset**. These are seizures that cannot be classified due to the lack of information on the onset. They may occur during sleep or under a condition in which witnesses cannot describe or record any kind of information (Fig.2.1).

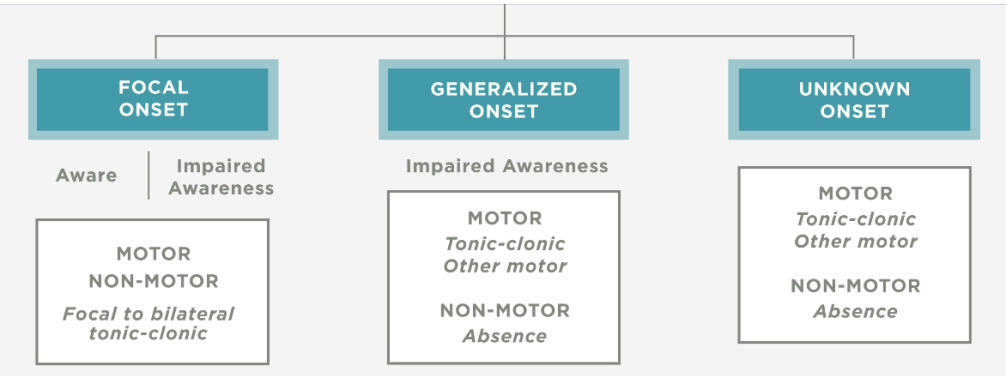


Figure 2.1: International League against Epilepsy 2017: Epilepsy Classification [8]

2.2.1 Focal seizures

Focal seizures originate in one hemisphere of the brain and can involve one or more areas of that hemisphere. They can also spread to the other hemisphere and become generalized seizures.
An **aura** can precede seizures. The aura is a set of physical, mental, or autonomic sensations that a patient feels before an attack. In focal seizures, symptoms depend on the area of the brain affected. Therefore, depending on whether the motor, sensory or language area is involved, there may be jerks, abnormal movements, sensory disturbances, and difficulty speaking. Visual, gustatory phenomena, behavioral alterations, sensations of strangeness, or déjà vu are also possible.
Focal seizures are classified according to the level of awareness as follows:

- Simple focal seizures: the person remains fully aware during the seizure;

- Complex focal seizures: the person experiences some loss of consciousness during the seizure.

Motor focal seizure can also be classified as

- Automatism: these are repetitive movements, such as rubbing the hands or licking the lips;
- Atonic: muscles of a specific area becoming limp or weak;
- Tonic: opposite of atonic in which the muscles, a body part or a limb become rigid;
- Clonic: repeated, rapid muscle contractions and relaxations. Usually affects arms and legs;
- Myoclonic: similar to clonic but last for less than a second, and they may occur in clusters. During clonic seizures, the patient may lose consciousness, while during myoclonic seizures vigilance is maintained.

Nonmotor focal seizures can instead be classified as

- Autonomic dysfunction: effects related to the autonomous system such as gastrointestinal sensations, feeling of heat or cold, Flushing, sexual excitement, piloerection and palpitations;
- Cognitive dysfunction: alteration of cognitive domains or symptoms such as déjà vu, hallucinations, illusions or perceptual distortions;
- Emotional dysfunction: manifested by sudden emotional changes or difficulties in controlling emotions;
- Sensory dysfunction: difficulties in receiving and responding to information that comes in through the senses leading to be oversensitive or undersensitive. This can affect one or more senses;
- Behavior arrest: cessation of movement and lack of responsiveness as the main feature of the entire seizure.

Focal seizures can evolve into a generalized one that causes loss of consciousness. This is called **secondary generalization** and occurs when a partial seizure spreads and activates the entire cerebral cortex. Activation can occur so quickly that the initial partial seizure is not clinically apparent or is very short.

2.2.2 Generalized seizures

Generalized seizures originate from abnormal electrical activity in both hemispheres. Awareness is often compromised by loss of consciousness. Generalized seizures can be classified as motor and nonmotor seizures. The classification is similar to the one for focal seizures, but here we also have the following.

- Generalized Tonic-Clonic Seizures, also known as **grand mal**, are a type of motor seizures divided into two phases: the tonic phase, characterized by muscle rigidity, and the clonic phase, characterized by moments of muscle relaxation. Consciousness is often lost with subsequent injury from a fall;

- Absence seizures, also known as **petit mal**, are a type of nonmotor seizure characterized by short and sudden loss of consciousness. They can be divided into typical and atypical seizures. **Typical** seizures are characterized by episodes of loss of consciousness that last a few seconds with blinking; muscle tone may or may not be lost. Patients do not fall or have convulsions. They suddenly interrupt the activity they are doing and then resume it, without symptoms or awareness that a seizure has occurred [12]. **Atypical** seizures differ from typical absence seizures because they last longer, spasms or automatic movements are more pronounced, and the loss of consciousness is less complete. They often occur in the context of **Lennox-Gastaut syndrome**.

2.3 Causes

The causes of epilepsy are different for each person, and most of the time the causes are not known. Doctors can trace epilepsy directly to genetics, brain trauma, autoimmune disorders, metabolic problems, or infectious diseases. Each cause has different signs, diagnoses, and treatment options.

- Genetics: Epilepsy is said to have a genetic cause if seizures are the result of a known or presumed genetic defect. Genetic epilepsy may or may not be inherited, as some changes in genes can spontaneously occur without being present in either parent. For most people, genes are only part of the cause of epilepsy. Certain genes may make a person more susceptible to environmental conditions that trigger seizures;

- Structural causes: These can be congenital or acquired. A *congenital cause* is a developmental change in the brain that a person is born with. Some congenital structural causes may also have a genetic component. These causes are often seen in babies born to women whose pregnancies have been otherwise normal (due to infections in the mother, poor nutrition, or oxygen deficiencies). An *acquired cause* is due to a process or injury that occurs in someone with a previously normal brain structure. Examples are brain tumors, strokes, or head traumas;

- Infections: Epilepsy is said to have an infectious cause if there is evidence of a brain infection that leads to seizures. Infection is probably the most common cause of epilepsy worldwide, but is more common in the developing world. Meningitis, HIV, viral encephalitis, and some parasitic infections can cause epilepsy.

- Autoimmune disorders: These are caused by a change in the immune function of the body. Our immune system protects the body from foreign substances or things that could harm the body, but there are conditions that cause your immune system to attack brain cells, leading to seizures;

- Metabolic causes: The body contains many enzymes that are responsible for breaking down the various parts of the food we eat to nourish the body. If there is underactivity or blockage in one of these enzymes, this can lead to problems metabolizing one of the components in food or problems generating energy to sustain the body's function, and then this can lead to seizures.

2.4 Treatment

Epilepsy has the tendency to repeat epileptic seizures, so while single-onset treatment should not be performed (except for particular cases), treatments must be carried out until complete control of seizures.
The first treatment step is usually to find the right medicine or Anti-Epileptic Drug (AED). AEDs can be needed indefinitely, and no drug is able to control all types of seizures, and different patients require different types of drugs depending on the type of seizure; thus, the first step is deciding whether the patient's seizures are partial or generalized. Most patients can be optimally managed with a single AED. One must be sure that a given drug has failed before moving on to an alternative drug or a two-drug combination. If the patient has persistent seizures, but no adverse effects, the dose can be increased as tolerated or until seizure control is obtained. Antiepileptic drugs can eventually be successfully withdrawn in more than 60% [22] of patients who remain free from seizures. Most neurologists require that patients remain seizure-free for 2 to 4 years before discontinuing AED, and drugs are generally discontinued over a 2 to 6 month period. Despite the availability of many anticonvulsants, they are only effective in controlling seizures in 7 out of 10 patients, leading to Drug resistent epilepsy.
Drug resistant epilepsy refers to a condition in which seizures in patients with epilepsy are not adequately controlled despite treatment with multiple antiepileptic drugs (AED). There is no single definition of drug-resistant epilepsy, as it can depend on the type of seizure and syndrome. In general, drug-resistant epilepsy is classified when a patient does not achieve seizure freedom for 12 months during long-term treatment with several appropriate AEDs at the maximum tolerated doses [22]. Up to one in three patients with epilepsy experience drug-resistant epilepsy.
Due to this, there are other approaches that have made it possible for most people to achieve seizure control that do not require AEDs. Such approaches include:

- Dietary therapy: is an approach to help control seizures, usually in conjunction with seizure medications. The most used one is the *ketogenic diet* special high-fat, low-carbohydrate diet, prescribed and monitored in the hospital;

- Surgery: is worth considering in patients in whom seizures and/or medication side effects significantly impair quality of life. Surgical treatment is indicated in such patients if seizures arise from an area that can be removed without causing unacceptable neurological deficits;
- Alternative therapies: treatments used instead of conventional medicine that are used together with AEDs. These therapies can include natural products such as herbs, vitamins, minerals, and probiotics, used in conjunction with diet therapy. Other practices may include acupuncture, meditation, relaxation techniques, massage therapy, yoga, and hypnotherapy;
- Devices: another option to control seizure is called *Neuromodulation*. This therapy involves using a device to send small electrical currents to the nervous system to change the electrical current in the brain to compensate for possible abnormal discharges that cause seizures. There are different neuromodulation approaches such as vagus nerve stimulation VNS, deep brain stimulation DBS and responsive neurostimulation RNS;
- Seizure alert devices: are electronic devices that can detect the onset of seizures and alert caregivers, health professionals, or clinicians about the incoming seizure, allowing them to provide timely assistance and put the patient into safety, helping to reduce the risk of injury and other complications.

In this context, this document presents research on AI methods that, using EEGs, could help monitor brain electrical activity, potentially allowing automatic prediction of epileptic seizures in advance, allowing patients to avoid dangerous situations or even plan the administration of preventive treatments, such as electrical stimulation or targeted drug delivery, thus improving their quality of life. It is important to note that AI-based seizure prediction is still in its early stages of development. However, the research presented in this document is promising and suggests that this technology has the potential to make a significant difference in the lives of people with epilepsy.

3 | Neural Networks Building Blocks

In this chapter, I introduce recurrent neural networks (RNNs), a type of artificial neural network suited for working with sequential data by extracting temporal features. After an overview of RNNs, I discuss long short-term memory (LSTM) and gated recurrent units (GRU), two specialized types of RNNs that can overcome some of the challenges of working and learning with long sequences of data. The last section, introduces convolutional neural networks (CNNs), a class of neural networks that specializes in processing data that has a grid-like topology.

3.1 Recurrent Neural Networks

Humans don't start their thinking from scratch every second. Our thoughts have persistence, allowing us to process a lot of information, grasp the important parts, and make our own thoughts about what we just witnessed. Traditional neural networks (feed forward network FFN) cannot do this because they are designed to treat each input as independent of the others. This means that they cannot learn long-term dependencies in sequential data, which is essential for many tasks such as natural language processing. Therefore, it is important to design models that can work with sequential data because observations in sequences have temporal patterns, meaning that previous observations have an effect on future ones. Persistence is one approach to learning such temporal patterns, and this can be achieved through memory. Recurrent neural networks (RNNs) were designed for this purpose.
What differentiates RNNs from FFNs is how information is passed through the network. While the FFN passes information through the network without cycles, the RNN has cycles that allow to transmit information back to itself. In figure 3.1 the neuron *'s'*, looks at some input x and outputs a value o. A loop allows information to be passed from one step of the network to the next.

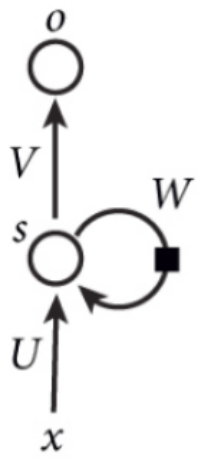


Figure 3.1: RNN schema with loops [14]

This can be treated as multiple copies of the same network, each of which passes the information processed at one time step to the next (Fig.3.2).

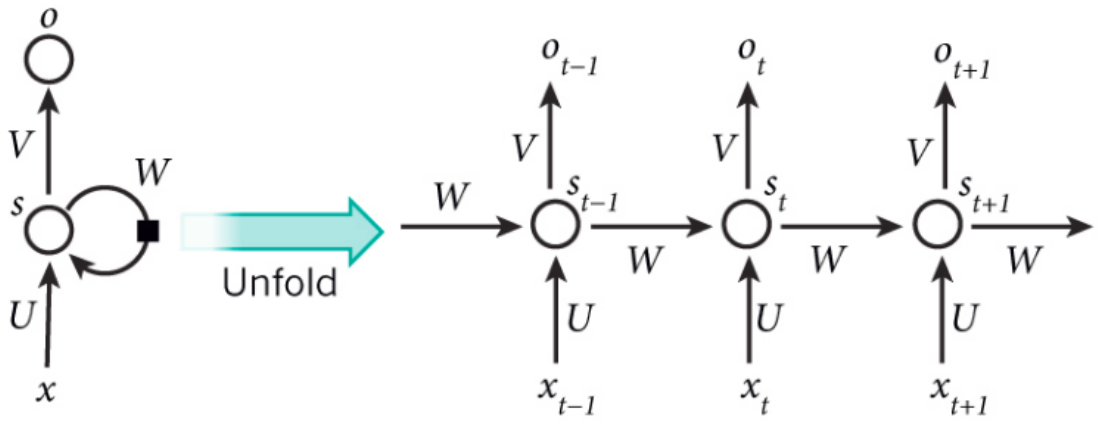


Figure 3.2: RNN schema unfolded in time [14]

This chain-like nature reveals that recurrent neural networks are related to sequences and therefore are the natural architecture of neural network to use for sequential data. This process of passing information can be described with the following mathematical formulas [9]:

$$h_t = \theta(Ux_t + Wh_{t-1} + b_t) \tag{3.1}$$

$$o_t = Vh_t \tag{3.2}$$

Where we have:

- n: number of samples;

- d: number of features for each sample;
- h: number of hidden units;
- t: timestamp being processed;
- x_t: the input at timestep t with dimension $\mathbb{R}^{n \times d}$;
- U: weight matrix from input to hidden state with dimension $\mathbb{R}^{d \times h}$;
- h_{t-1}: the hidden state computed at timestep $t-1$ with dimension $\mathbb{R}^{n \times h}$;
- W: weight matrix from hidden to hidden with dimension $\mathbb{R}^{h \times h}$;
- b_t: bias parameter at timestep t with dimension $\mathbb{R}^{1 \times h}$.

The activation function θ prepares the gradient for backpropagation, which is used to compute the hidden state h_t at timestep t, which has dimension $\mathbb{R}^{n \times h}$. This new hidden state h_t is then combined with its weights V with dimensions $\mathbb{R}^{h \times k}$ (where k is the number of classes) to generate the final output o_t. To correctly propagate the RNN error throughout the model, we use a technique called backpropagation through time.

3.1.1 Backpropagation Through Time

Backpropagation is a supervised algorithm used to update a model's weights to minimize the error of the predicted output compared to the expected output. It consists of four main repeated steps:

1. Present an input data and propagate it through the model (*forward pass*) to get a result;
2. Calculate the error by comparing the predicted result with the expected output (*supervised learning*);
3. Compute the derivatives of the error with respect to the network's weights;
4. Adjust the weights (*backward pass*) to minimize the error.

This procedure is well suited for feed-forward networks where the inputs and outputs are fixed in size.
Backpropagation through time (BPTT) is an adaptation of backpropagation to RNN. It works by unfolding the RNN into a traditional feed-forward neural network, in which we can apply backpropagation (just like figure 3.2). Unfolding means replicating the network for each data in the sequence. This allows us to adapt backpropagation to RNN, since the weights associated with the input x (denoted by U) and the previous step h_{t-1} (denoted by W) are shared. This avoids an explosion of parameters due to the unfolding in time of the network. Other advantages of this approach are: (1) we can work with sequences of any length and (2) since the

weights are shared, the model does not become too complex, helping in controlling overfitting. Conceptually, BPTT works by unrolling all input timesteps. Each timestep has one input x_t, one copy of the network s_t, and one output o_t. Errors are then calculated and accumulated for each timestep t. The network is rolled back up and the weights are updated. Each timestep of the unrolled recurrent neural network may be seen as an additional layer of the model, which gives the order of dependencies of the problem that is managed by the hidden state h of the previous timestep and is taken as input in the subsequent timestep.

3.1.2 Problem of Long-Term Dependencies

The main problem with the BPTT algorithm arises when the model is fed with data from very long sequences.If the input sequences are made up of thousands of timesteps, then this will be the number of derivatives required for a single weight update. In this case, the gradients accumulate and the method becomes unstable. Eigenvalues less than 1 cause the gradients to decrease with each layer until they vanish. This practically makes the contribution of further timesteps meaningless with respect to the current one. This can cause the network to become stuck in a local minimum, or even to fail to converge to any solution. However, if the eigenvalues are greater than 1, the gradients increase with each layer, causing an explosion. In practice, this means that the weights of the network are updated by very large amounts, which can cause the network to overshoot the optimal solution and even cause the network to diverge, meaning that the weights become infinitely large.

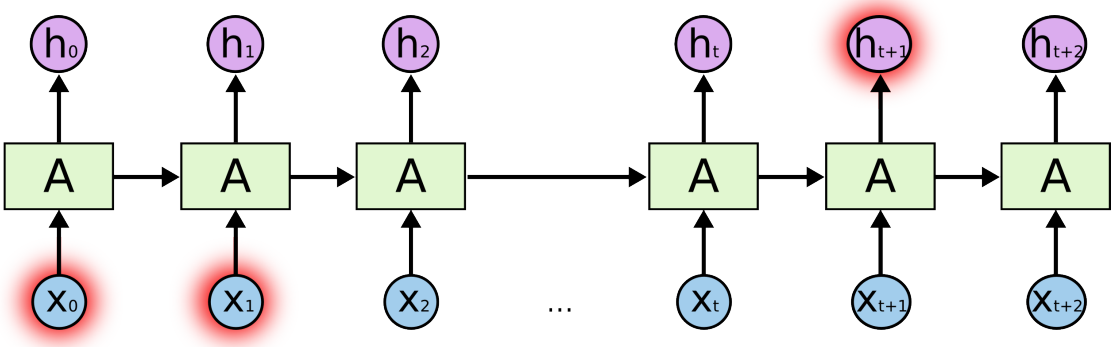


Figure 3.3: Long-term dependency problem: the network cannot learn from distant timesteps [14]

3.1.3 Truncated Backpropagation Through Time

To address the issue of having a computationally expensive algorithm, a variant of BPTT has been developed.
The main idea of this variant is to stop computing the sum of gradients after timestep k_1 and perform the update back for a fixed number of timesteps k_2. This

leads to an approximation of the true gradients and gives quite good results in practice. This version is called trucated backpropagation through time (TBPTT). This algorithm introduces two new variables:

- k_1: the number of forward passes before a new update of the weights. It controls how often the weights are updated and therefore how fast or slow the training will be;

- k_2: the number of timesteps in which we apply BPTT. It should be large enough to capture the temporal dependencies, but not too large to avoid vanishing gradients.

Figure 3.4 shows graphically the differences between the weight update that occurred in backpropagation through time and the truncated backpropagation through time.

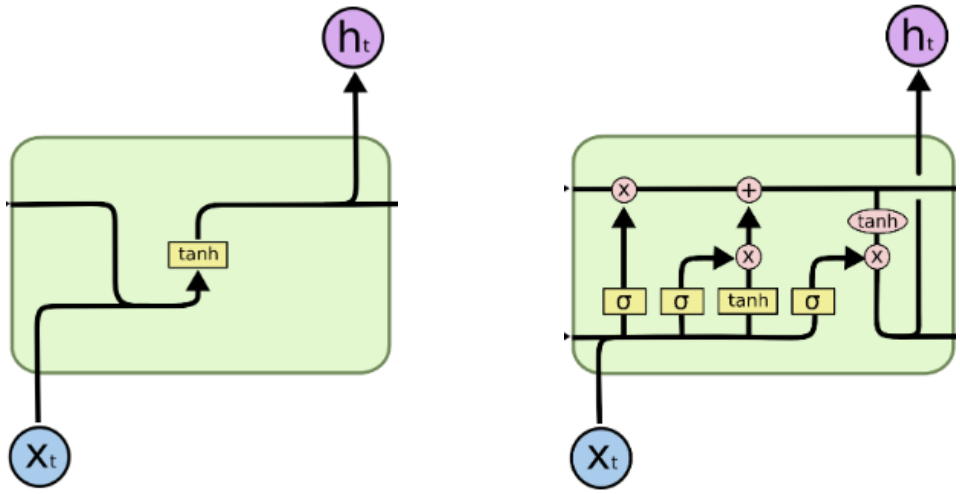
(a) BPTT **(b)** TBPTT

Figure 3.4: Graphical representation of BPTT and TBPTT.
Blue arrows represent the forward-pass step, and red arrows represent the backward-pass step. Dots represents the internal state reset [5]

The advantage of having a less computationally expensive algorithm is balanced by the fact that the algorithm is an approximation that might not work well for sequences where the dependencies can be further apart from each other.
A better solution to this problem and the vanishing and exploding gradients one is to use networks that are specifically designed to learn long-term dependencies, such as long short-term mermory (LSTM [10]) and gated recurrent unit (GRU [6]) networks.

3.1.4 Fundamentals of LSTM

Long short-term memory (LSTM) networks are a special type of recurrent neural network (RNN) designed to learn long-term dependencies by overcoming the problems of RNN. The main difference between RNNs and LSTMs is that LSTMs have a more sophisticated internal structure that incorporates mechanisms to handle the memory of past information (Fig. 3.5).



(a) RNN **(b)** LSTM

Figure 3.5: Difference between the internal structure between RNN and LSTM [14]

The LSTM is composed by four different submodules:

- input gate: controls how much of the current input is added to the cell state;
- forget gate: controls how much of the previous cell state is forgotten;
- cell state: is the memory of the LSTM cell. It stores information about the past inputs;
- output gate: controls how much of the cell state is outputted as the hidden state.

The key component that allows LSTM to learn long-term dependencies is the cell state (Fig. 3.6).

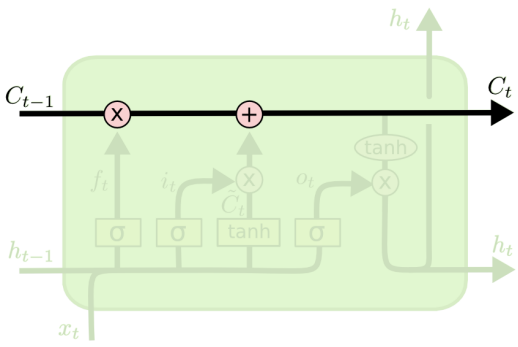


Figure 3.6: LSTM cell state highlight [14]

This component is regulated by *gates* (Fig. 3.7) that allow information to be stored or deleted from memory. These gates are learned during training, so they can adjust according to how much information of the past to maintain or not.

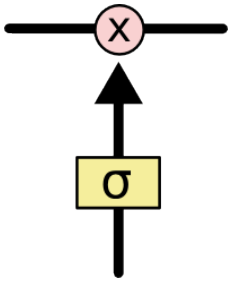
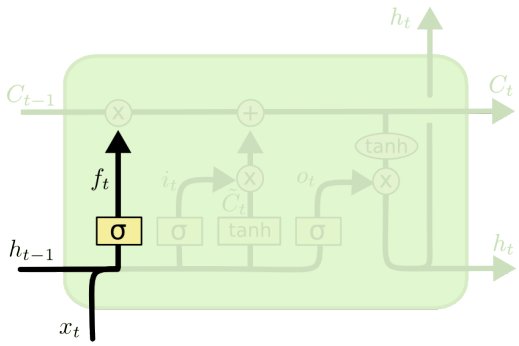


Figure 3.7: LSTM gates structure [14]

3.1.4.1 Step-by-step walkthrough

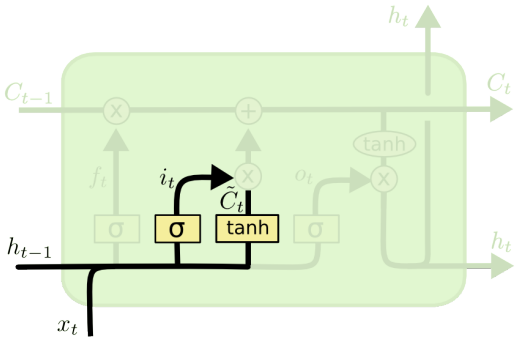
At each new data sample, the first step (Fig. 3.8) is to decide how much information to remove from the cell state. This decision is made by the sigmoid layer called *forget gate*. This layer takes as input the hidden state at previous timestep h_{t-1} and the current input x_t and returns a value between 0 and 1. A value of 0 means that we want to remove all past memory, while a value of 1 means that we want to keep all of it. These values are stored in f_t.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure 3.8: LSTM step by step: forget past information [14]

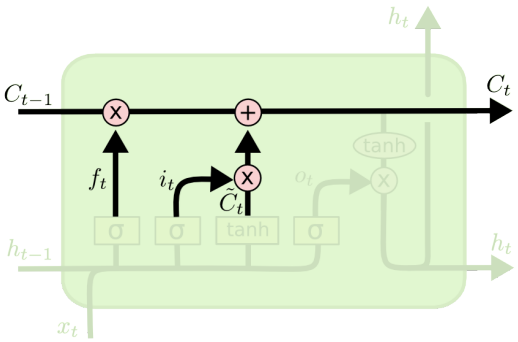
The next step (Fig. 3.9) is deciding what new information at timestep t we want to store in the cell state. First, we find the relevant information in x_t creating a vector of new candidate information to store ($\tilde{C}_t$). The second step is to compute, through the *input gate layer*, which information to update. This is done using a *sigmoid* function which values in (0, 1) measure how relevant or irrelevant the information at timestep t is. These relevancy scores are stored in i_t.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure 3.9: LSTM step by step: store new information [14]

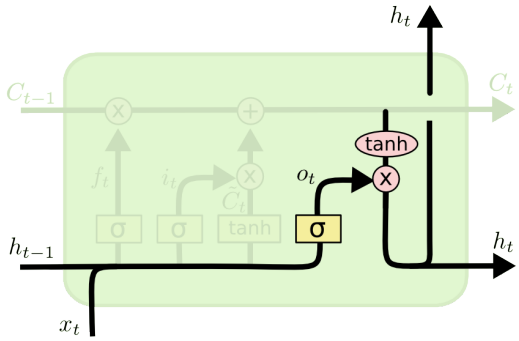
In the next step (Fig. 3.10) we want to update the previous cell state C_{t-1} (called *long-term memory*) to create the new cell state C_t. This is done by multiplying the old cell state C_{t-1} with f_t, forgetting the past information that is not relevant to the information in x_t. Then we add the new relevant information by multiplying by $i * \tilde{C}_t$.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 3.10: LSTM step by step: cell state update[14]

In the final step (Fig. 3.11), we are going to decide which information to output from the LSTM cell. First, we create the output o_t by combining the previous hidden state h_{t-1} and the new input x_t with a *sigmoid* function. Then, we pass the cell state C_t through *tanh* and multiply it by the output of the sigmoid gate, so that we only output the most relevant parts for timestep t in h_t (called *short-term memory*).



$$o_t = \sigma\left(W_o\ [\,h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 3.11: LSTM step by step: cell output[14]

The definition of the cell state C_t and the use of gate mechanisms allow LSTMs to solve the vanishing gradient problem by controlling how much of the old state is forgotten and how much of the new data is added. This keeps the gradients under control, allowing LSTMs to learn long-term dependencies in the data.

3.1.5 LSTM drawbacks and variants

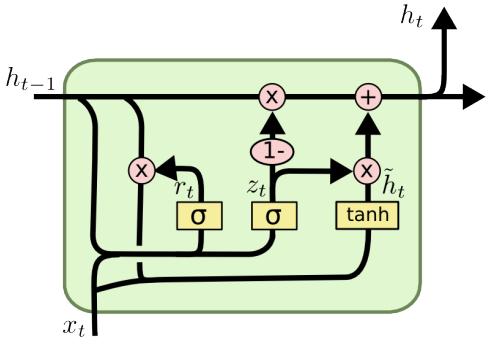
LSTM networks are powerful in learning long-term dependencies, but they suffer from some drawbacks because, to handle long sequences, they are quite complex. Some problems of the LSTMs are the following:

- Computational complexity: LSTM networks are computationally expensive to train and run. This is because they require more memory and more computations at each time step than other types of neural networks. This is mainly because all the gates need to be trained to learn how much information to maintain, forget, and pass through the subsequent time steps.
- Overfitting: LSTM networks are prone to overfitting, especially when training data is limited or noisy. This is because they have many parameters and a very complex architecture. To train effectively, they require large amounts of data, which is not always possible to obtain.
- Sensitivity to hyperparameters: LSTM networks are sensitive to hyperparameters, such as learning rate and number of epochs. This means that it can be difficult to find the right hyperparameter settings for a given problem, leading to long optimization times.

For these reasons, variants of LSTMs were studied. One such variant is called gated recurrent unit (GRU in Fig. 3.12). This is a simplified version of the LSTM in which the forget and input gates are combined into a single *update gate*. This variant is then composed of only two gates:

- update gate controls how much of the previous hidden state is combined with the current input to produce the new hidden state (z_t);

- reset gate controls how much of the previous hidden state is forgotten (r_t).



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure 3.12: GRU internal structure[14]

Having only two gates leads to a simpler internal structure, resulting in faster training and inference steps.
Compared to LSTM networks, GRU networks are less powerful, but are also simpler and faster to train and run.

3.2 Convolutional Neural Networks

A convolutional neural network, also known as CNN, is a class of neural networks that specializes in processing data with grid-like shapes. Each neuron of the human brain works in its own receptive field and is connected to other neurons, so it covers the entire visual field. Just as each neuron responds to stimuli only in the restricted region of the visual field called the *receptive field*, each neuron in a CNN processes data only in its receptive field as well. These neurons are arranged so that they detect simpler patterns first and more complex patterns later on. At its basic level, CNN can be thought of as a kind of neural network that uses many identical copies of the same neuron, allowing the network to express computationally large models while keeping the number of parameters small.

3.2.1 Convolutional Layers

The convolutional layer (Conv Layer) is the main building block of a CNN. Suppose that we have some data samples that we want to classify based on some local properties of the data. The CNN approach involves (Fig. 3.13) the usage of neurons (**A**) that look at small *time segments* of our data (analogous to the receptive field) and compute some features from it. These features, which are the output of the *convolutional layer*, are then passed through a fully connected layer (**F**) for the final classification. One useful property of convolutional layers is that they are *composable*: you can feed the features of a previous convolutional layer as input of another to extract more relevant and high level features (Fig. 3.14).

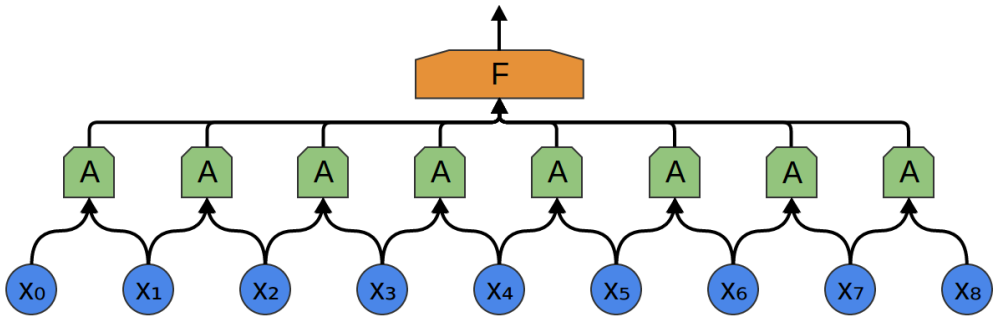


Figure 3.13: CNN simple structure: the conv layer look at two timesteps but it can be expanded to have a greater receptive field and look at more timesteps at once [13]

Between one convolutional layer and another, there is often a *pooling layer* (Fig.

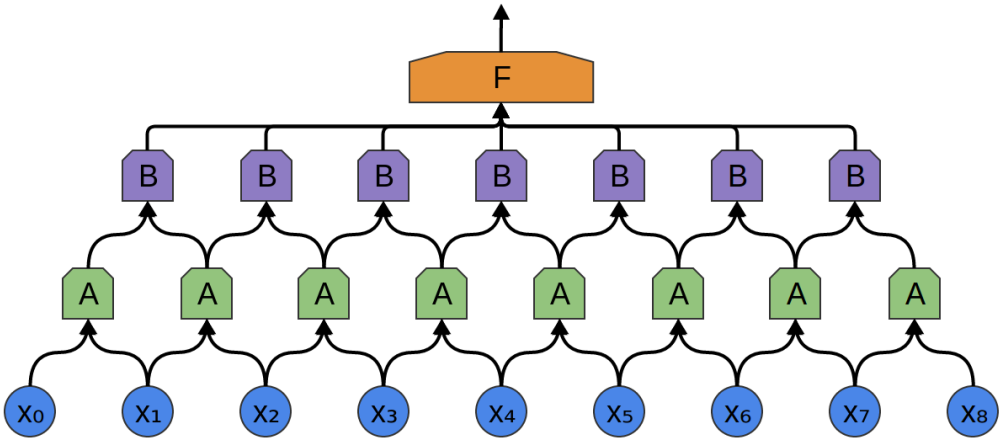


Figure 3.14: CNN composition: the output of the previous layers (**A**) is the input of the second layer (**B**). This can be used to extract high-level features [13]

3.15). The pooling layer replaces the network output at certain locations by deriving a summary statistic of the nearby outputs. This helps for two main reasons:

- reduces the number of parameters to learn: by reducing the spatial dimensions of the features, pooling layers can reduce the number of parameters that need to be learned in the network. This can make the network faster to train and more efficient;
- makes the network more invariant to translation: pooling layers can help to make the network more invariant to small changes in the position of features in the input. This is because pooling layers summarize the features in a region of the input rather than focusing on individual samples.

The most used pooling layer is the *max-pooling layer* which takes the maximum of features over blocks of a previous layer. It tells us if a feature was present in a region of the previous layer, but not where. Convolutional neural networks can work on data of any dimensionality, but the most common use is to apply 2D convolutional neural networks to images or video sequences.

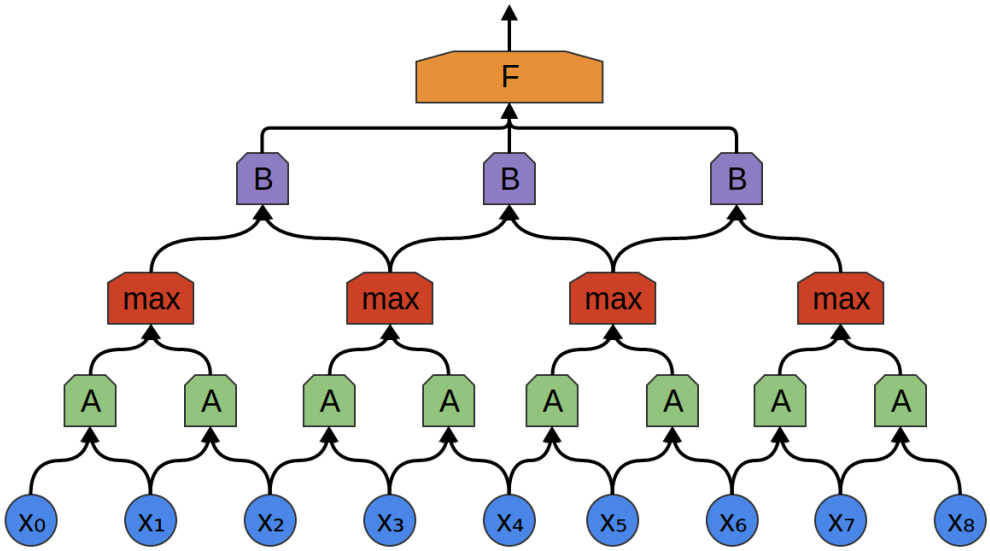


Figure 3.15: CNN schema: between the conv layer **A** and **B** we have the pooling layer **max** [13]

In these examples, I introduced 1D convolutional layers to extract temporal features from data sequences that do not need to be images, such as EEG signals. These blocks were used as part of one of the two models proposed in chapter 5.

3.2.1.1 Into convolution

What does a convolutional layer exactly compute?
Intuitively, this layer performs a dot product between two matrices, where one matrix is the set of learnable parameters known as a *kernel*, and the other matrix is the input data. The convolution operation is performed by sliding the kernel over the input signal and multiplying the values of the kernel and the input signal at each point. The results of these multiplications are then summed together to produce a single value at each point in the output signal. Here, an example:

```
Kernel = [3, 2, 1]
Input = [1, 2, 3, 4, 5]
```

to have a better feeling of the slide of the kernel I flip the input:

```
Kernel = [3, 2, 1]
Input = [5, 4, 3, 2, 1]

Step 1:                 Step 2:                 Step 3:
        3 2 1                   3 2 1                   3 2 1
5 4 3 2 1               5 4 3 2 1               5 4 3 2 1
---------               ---------               ---------
```

```
        3   = 3              6 2 = 8              9 4 1 = 14

Step 4:                  Step 5:                  Step 6:
  3 2 1                  3 2 1                    2 1
5 4 3 2 1                5 4 3 2 1                5 4 3 2 1
---------                ---------                ---------
  12 6 2    = 20         15 8 3      = 26         10 4        = 14


Step 7:
1
5 4 3 2 1
---------
5           = 5
```

The result of the convolution is then the new matrix [3, 8, 14, 20, 26, 14, 5].
The kernel can be thought of as a window that slides over the input and the output of the dot product is then used to update the neuron activation. The *receptive field* of a neuron is then defined as the input area covered by the kernel when the neuron is activated.
From this intuitive explanation, we could now define the mathematical formula of convolution:

$$(f * g)(t) := \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau$$

that can be read as: to convolve $(*)$ a kernel (f) with an input signal (g) we must: flip the signal $(-\tau)$, slide for the desired time (t) and accumulate every interaction $(\int)$ with the kernel (f) [7] [15].
Recalling figure 3.15 we have that the convolutional layer $(\mathbf{A})$ consists of weights that describe its behavior. Negative weights inhibit neurons from firing, whereas positive weights encourage them. Convolution handles the wiring of neurons, describing all the weights and which ones are identical. During training, the layer learns different weight configurations that the kernel uses to extract different temporal features from the input signals.

One thing to note is that the kernel is shared in a convolutional layer, allowing the same set of weights to be used to convolve multiple inputs. This has several advantages:

- reduces the number of parameters in the network, making it more efficient to train;
- helps to extract more general features from the input data, because the kernel is able to learn from multiple inputs at once;

- helps to reduce overfitting, because the kernel is not being trained on a specific set of inputs and can generalize among different ones.

4 | Problem definition and experimental setup

In this chapter, I define the problem of epileptic seizure forecasting and explain why it is a concern. After defining the problem, I introduce the dataset used in this study, along with its annotations and preprocessing pipeline. Finally, I present the various experiments conducted, along with the metrics used to evaluate the performances of the models that proposed in Chapter 5.

4.1 Problem definition

As stated in chapter 2 epilepsy is a severe neurological disorder that leads to recurrent seizures. Although antiepileptic drugs can reduce clinical complications and mortality rates, 30% of patients are refractory to such medications [22], leading to the development of alternative treatments. The unpredictability of seizures significantly affects the patient's quality of life due to the risk of injury, mortality, and psychosocial disability.
Evidence suggests that specific alterations and patterns can be observed in brain dynamics before epileptic attacks [3]. This finding inspired the building of devices capable of anticipating seizures by analyzing electroencephalogram EEG. A reliable monitoring device would allow patients to avoid dangerous situations or even plan the administration of preventive treatments, such as electrical stimulation or targeted drug delivery, thus significantly improving their quality of life.
In such a setting, seizure *forecasting* aims to anticipate an upcoming seizure before it clinically happens. This is a challenging task because finding patterns in EEG seizures' manifestations vary between patients (based on their psycho-physical characteristics and clinical history) and between the types of seizures with which a patient is affected.
The most recent approaches to this problem exploit deep learning techniques to extract nonlinear features directly from the raw EEG signals and then use these to train the models.
In this research, I feed LSTM and GRU with EEG signals, allowing the models to exploit interchannel dependencies to extract temporal features to perform the prediction.

C3	C4	Cz	F3	F4	F7	F8	Fp1	Fp2	Fz
O1	O2	Oz	P3	P4	P7	P8	Pz	T7	T8

Table 4.1: List of channels used in the EEG dataset

4.2 Dataset description

The data used are a set of continuous long-term multichannel EEG recordings collected by the Epilepsy and Clinical Neurophysiology Unit at the Eugenio Medea IRCCS Hospital in Conegliano, Italy. Data were recorded from 29 epileptic patients (15 male and 14 female, referred to as Patient 1 through Patient 29 for privacy reasons) at a sampling rate of 256 Hz. I used 20 common channels based on the international standard 10-20 EEG scalp electrode positioning system (Fig. 4.1). The channels used are reported in table 4.1.

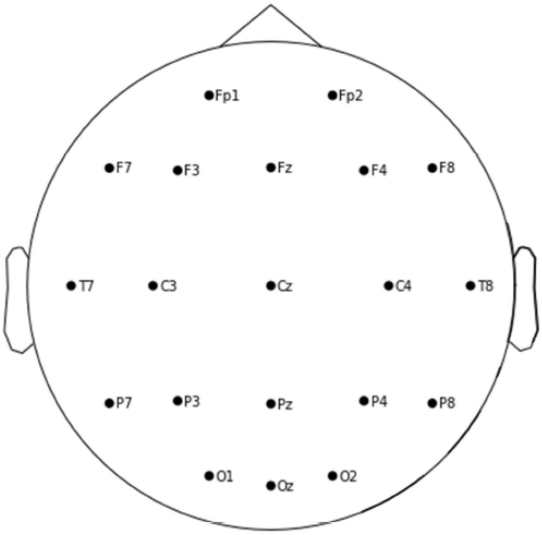


Figure 4.1: Scalp positioning of the 20 common EEG used in the dataset [23]

4.2.1 Data labeling

The EEG signal recorded from epileptic patients can be categorized into four stages:

1. interictal state: is a period of regular brain activities between two consecutive seizures (approximately 30 to 120 minutes preceding a seizure);
2. preictal state: refer to the period included between approximately 30 minutes before a seizure onset;
3. ictal state: the period in which the seizure occurs;

4. postictal state: the period immediately following a seizure for a few minutes.

In the data, the onset and end of the ictal states were manually identified using video recorded data from video-EEG monitoring by two expert clinicians. Figure 4.2 shows how an EEG signal is segmented into the four categories above.

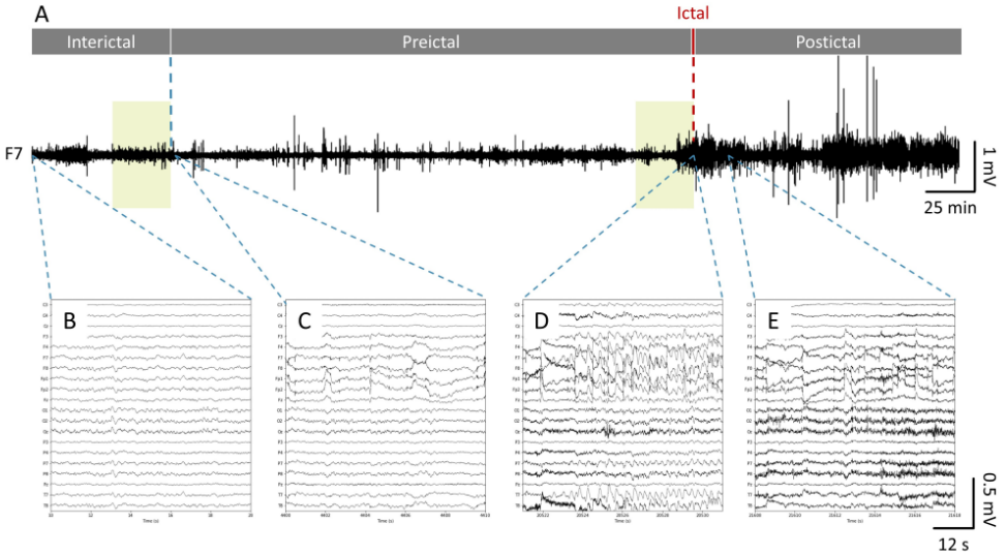


Figure 4.2: Segmentation of EEG recording: (**A**) trace depicting a recording from a single EEG channel during a seizure. It is divided into: interictal, preictal, ictal, and postictal stages. The highlighted areas represent the 30 minutes before the seizure from the preictal and interictal states used for the prediction task. (**B**),(**C**), (**D**) and (**E**) are magnification of the recordings from 20 channels at the beginning of each stage [23]

The goal of the seizure prediction task is to discriminate between the preictal state immediately preceding a seizure and a general signal recorded during the interictal state. For this, I used two binaries categories based on the distance to the upcoming seizure:

- class 0: are signals sampled from the time window between 0 and 30 minutes before the seizure (*preictal class*);
- class 1: are signals sampled from 30 minutes randomly selected from the interictal state (*interictal class*).

To guarantee the use of appropriate interictal time periods, we excluded seizures that occurred in recordings shorter than 3 hours, resulting in a total of 93 seizures retained as valid data.

4.2.2 Preprocessing steps

Several filtering procedures were applied to improve signal quality. Specifically, a 125 Hz low-pass filter and a 1 Hz high-pass filter were applied to retain high-frequency

Frequency	Overlapping	#Segments	#Channels	Seq.Len
128	0	57.612	20	640
128	3	143.940	20	640
64	0	57.612	20	320
64	3	143.940	20	320

Table 4.2: Dataset used: 5 second segments variants

signals relevant to abnormal brain activities before seizures, while simultaneously removing DC offset and baseline fluctuations. Furthermore, to mitigate power line interference, two notch filters operating at 50 Hz and 100 Hz were incorporated into the preprocessing pipeline. Finally, the EEG signals were normalized by subtracting the average EEG reference computed in the training data of each patient and balanced to have an equal number of preictal and interictal segments. Depending on the experiment, the EEG signals were downsampled at 128 or 64 Hz and segmented into 4-, 5-, 10-, 20- or 30-second intervals.
Based on the evaluation metrics in Section 4.3, the best performing dataset is composed of 5 second segments, and, therefore, experiments were carried out making the segments overlap by half (rounded up) of their length (for a total of 3 second overlap). In total, I ended up having four different datasets depicted in table 4.2.

4.3 Evaluation criteria

Evaluation criteria are essential to evaluate the performance of models throughout their life cycle. The following metrics were used in both optimization and training and allowed me to design the best models used in the experiments that I am going to present in chapter 5.
I have used a total of three metrics: *accuracy*, *specificity*, and *sensitivity* each of these computed from:

- TP (True Positive): The number of positive instances that the model correctly predicted as positive;
- TN (True Negative): The number of negative instances that the model correctly predicted as negative;
- FP (False Positive): The number of negative instances that the model incorrectly predicted as positive;
- FN (False Negative): The number of positive instances that the model incorrectly predicted as negative.

4.3.1 Accuracy

It is the most common metric used to evaluate models. Measure the percentage of predictions that the model makes correctly on the total number of predictions. It is calculated as:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FN + FP)}$$

It is a simple and intuitive metric and is easy to calculate and interpret. However, it is not always the best metric for evaluating deep learning models. If a dataset is imbalanced, meaning that there are many more examples of one class than another, an accuracy metric can be misleading. In this case, a model could achieve high accuracy simply by always predicting the majority class, even if it is making many incorrect predictions for the minority class. For this reason, this is not the only metric used, even if it is a good starting point for evaluating a model's overall performance.

4.3.2 Specificity

Specificity is a metric that measures how well a deep learning model is able to identify negative examples. It is calculated as the proportion of true negatives to the total number of negative examples:

$$Specificity = \frac{TN}{(TN + FP)}$$

A high specificity indicates that the model is good at avoiding false positives.
Specificity is an important metric to consider when evaluating deep learning models for tasks where it is critical to avoid misclassifying negative examples. For example, in a medical diagnosis system, a false positive (predicting that a patient is sick when they are actually healthy) could lead to unnecessary and potentially harmful interventions. However, specificity can be confusing because it is possible to achieve high specificity simply by predicting the negative class for all examples.

4.3.3 Sensitivity or Recall

Sensitivity, also known as recall, is a metric that measures how well a deep learning model is able to identify positive examples. It is the opposite of specificity and is calculated as the proportion between the correct positive predictions and all positive predictions:

$$Sensitivity = \frac{TP}{(TP + FN)}$$

A high sensitivity indicates that the model is good at finding positive examples, even if it means that it also makes some false positive predictions.
Recall is an important metric to consider when evaluating deep learning models for tasks where it is critical to avoid missing positive examples. For example, in a

medical diagnosis system, a false negative (predicting that a patient is healthy when they are actually sick) could lead to a delay in diagnosis and treatment. However, the sensitivity can be confusing because it is possible to achieve high sensitivity by simply predicting the positive class for all examples. For this reason, sensitivity and specificity were used in conjunction to evaluate the performance of the model in both the optimization and training procedures.

4.4 Experimental approaches

A series of experiments and evaluations were conducted to measure the predictive capabilities of proposed models for the prediction of epileptic seizures. These experiments measured different aspects of the performance of the models. The experiments are as follows:

- Randomized cross validation (RCV): classic approach in which, multiple times, the data is split into training and test to evaluate the predictive capabilities of the model as the average performance. This method has been used to establish a baseline for the predictive capabilities of the model, from which all other approaches can then be compared;
- Leave one patient out (LOO): approach where the data is divided into 29 folds, each representing a single patient. The model is trained on all but one fold and then evaluated on the remaining one. This is the most challenging task to have good results with;
- Permutation experiments: RCV and LOO were repeated by randomly ordering the channels (Table 4.1) that are given as input to the models. This type of experiment aims to understand how the models exploit the relationships between channels, or if they are exploited as a whole, making the order of the channels irrelevant;
- Feature selection: three different feature selection methods were studied and implemented to evaluate whether all 20 channels were actually used during the training phase or if some contained redundant/irrelevant information. The implemented methods include two filter methods and one wrapper method.

4.4.1 Randomized Cross Validation (RCV)

Randomized cross-validation (Fig. 4.3) is a technique that aims to evaluate the generalizability of models by evaluating their predictive capabilities on data that were not used during training. This is a widely used technique for both the evaluation phase of a trained model and the optimization phase of hyperparameters due to its simplicity of implementation.
The main reasons for its use are:

- avoid overfitting, which occurs when a model learns too much from the training data and is unable to generalize. In this case, training the model on different data subsets helps it find patterns that help the model to predict unseen data;

- compute better generalization estimates, as the final performance of the model is calculated as the average of the training performed on different subsets.

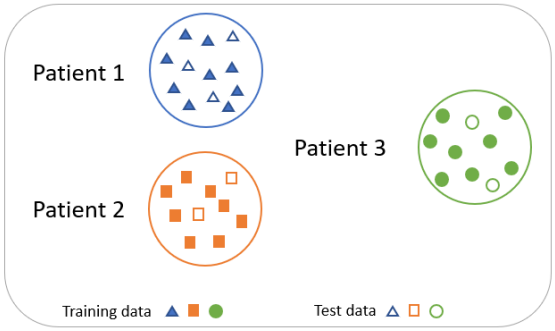


Figure 4.3: Randomized Cross Validation (RCV) schema

In this project, I used 5-fold cross-validation to evaluate the performance of the models. I randomly divided the data into five folds of equal size. For each fold, we trained the model on four folds and tested it on the remaining fold. We repeated this process until each fold was used for both training and testing. Finally, I computed the mean accuracy, sensitivity, and specificity across the five folds to obtain the final performance of the models.
The same experiment was then repeated by randomizing the order of the 20 channels within the dataset. The goal of this experiment was to see how the models exploit the information in the data, considering or not the order in which the channels are processed. This experiment is important to understand whether it is necessary to record EEG in a specific order to achieve better performance or whether the model is able to extract temporal features independently of the order, simplifying the data collection process.

4.4.2 Leave One patient Out (LOO)

Leave One patient Out (Fig. 4.4) is a performance evaluation technique in which a training dataset is created from 28 patients and a test dataset is created from the remaining patient on which the performance is evaluated. This procedure is repeated 29 times so that each patient is individually evaluated in the test dataset. Performances are evaluated for each patient, as well as the average of all results of individual patients.

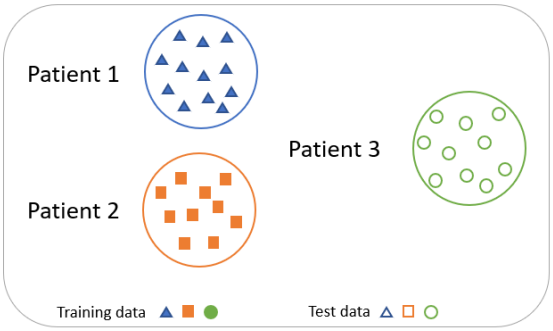


Figure 4.4: Leave One patient Out (LOO) schema

With this procedure, achieving good results is difficult because it aims to predict the onset of seizures in a patient based on seizures of other patients who, as described in Section 4.1, may have psychophysical characteristics or types of epileptic seizures that are completely different from those of the patient under test.
Even in this case, the experiments were repeated by randomizing the order of the channels in the training data.

4.4.3 Feature selection

Feature selection is a process of selecting input variables from a dataset. Typically, it involves selecting the most informative variables, thus reducing the dimensionality of the original dataset. In my case, the feature selection process aims to reduce the number of channels (Table 4.1) to a subset that retains only the most informative and useful channels for predicting an epileptic seizure (dimensionality reduction). This process is particularly important for three main reasons:

- it reduces the computational complexity of every activity that is done on EEGs;
- it can potentially improve performances and reduce the risk of overfitting by eliminating channels that do not contain useful information;
- a smaller amount of data therefore implies less setup time for models and the entire training and testing process

Feature selection follows these steps:

1. subset generation;
2. subset evaluation;
3. matching the stopping criterion? yes: return the results for the subset, no: go back to step 1)

Feature selection methodologies vary in how they generate and evaluate candidate subsets. The evaluation methods used in this research are *filter* and *wrapper* methods.

4.4.3.1 Filter methods

Filter methods apply a statistical measure to assign a score to each channel. The channels are then ranked by the score and either selected to be kept or removed from the dataset. Filtering techniques have some advantages, among them high speed, independence from the classifier, and scalability, but they suffer from low accuracy, since they consider each channel independently from all the others.
For EEG seizure prediction, the signal statistics used for channel selection is *variance* and it has been used in two different ways [18].
The first method estimates the plain variance of the preictal data of all available channels with the equation:

$$V_{preict}(c) = \frac{1}{k}\sum_{i=1}^{k}(x_c(i) - \mu_c^2)$$

where:

- $V_{preict}(c)$: variance of preictal data from channel c;
- k: total number of segments;
- x_c: preictal data coming from channel c;
- μ_c: mean of the data from channel c.

The channels are then ordered by variance and several experiments were performed, selecting the first N channels in descending order.
The second method estimates the difference between the variance of the preictal data and the interictal data of all channels with the equation:

$$V_{diff}(c) = V_{preict}(c) - V_{interict}(c)$$

The N channels with the highest difference between the preictal and interictal variance are then selected to perform several experiments.

4.4.3.2 Wrapper methods

Wrapper techniques use a classification algorithm to evaluate candidate channel subsets. Evaluation is obtained by training and testing models using the selected subset of channels as input. Since they require multiple training and test phases, wrapper techniques are more computationally expensive than other feature selection methods. However, they can achieve better results and more accurately estimate the most informative candidate channel subsets.
The following algorithm is inspired by the one proposed in [11]:

```
channels                # the 20 channels
dataset                 # dataset with 20 channels
baselineAcc = 0         # baseline accuracy
hasRemoved = True       # stop criterion

baselineAcc = train_model(dataset)

while hasRemoved:
    hasRemoved = False
    candidate = {}

    for toRemove in channels:
        newSubset = remove(channels, toRemove)
        subsetDataset = get_data_subset(dataset, newSubset)

        tempAcc = train_model(subsetDataset)
        candidate[toRemove] = tempAcc

    bestCandidate, bestAccuracy = select_candidate(candidate)
    if(bestAccuracy >= baselineAcc):
        channels = remove(channels, bestCandidate)
        hasRemoved = True
```

Listing 4.1: Wrapper algorithm for channel reduction

This algorithm is quite simple.
After calculating the baseline accuracy (*baselineAcc*), it starts by removing one channel at a time to create a subset of data consisting of 19 channels. For each subset, the accuracy is calculated and saved in a dictionary, along with the name of the channel removed.
When all possible subsets with 19 channels were calculated and saved in the dictionary, the one with the best accuracy is retrieved along with the associated removed channel (*bestCandidate* and *bestAccuracy*). If this *bestAccuracy* is not worse than the baseline, then it means that it is possible to remove the *bestCandidate* channel without worsening the performance.
The channel is then removed, and the flag *hasRemoved* is set to *true*. This means that the algorithm, within the while loop, will start to evaluate the channels again, but this time with the 19 that remained from the previous iteration.
This procedure continues until it is possible to find channels to remove that do not worsen the performance compared to the baseline. If this does not happen, then the flag *hasRemoved* will remain *false*, resulting in the exit from the while loop and the interruption of the procedure.
This is computationally expensive and time-consuming because it consists of two

loops, one nested into the other, and for each new subset it is necessary to retrain the model from scratch. Despite this, this procedure results in a more accurate selection of channels than filter methods.

5 | Models and Optimization

In this chapter, I present the two different architectures that were used experiment in the epileptic seizure prediction problem.
The first model uses only LSTM to extract temporal features from raw EEG signals and to find patterns and correlations between these features.
The second model, on the other hand, is a more complex architecture that combines the advantages of CNN to extract temporal features from raw data and the characteristics of GRUs to learn temporal correlations between these features. For both models, the final architectures that were obtained after optimizing the hyperparameters on each dataset used in the experiments are presented.

5.1 Optimization

Hyperparameters are parameters that control the training process of a deep learning model. They are not learned from the data, but are set by the user before training begins. These kinds of parameters are particularly important because:

- they determine the architecture of the model. Hyperparameters such as the number of layers and neurons in each layer play a key role in determining the complexity and expressiveness of the model;
- they control how the model learns. Hyperparameters such as the learning rate control how the model updates its parameters during training;
- they can help to prevent overfitting. Hyperparameters, such as weight decay or dropout values, can help to prevent the model from learning the training data too well and from being unable to generalize to new data.

In general, they can be divided into two categories. *Architectural hyperparameters* determine the structure of the model, such as the number of hidden layers or neurons inside each hidden layer. *Learning hyperparameters* instead control the training process, such as the learning rate.
Since they have a direct impact on the model's performances, finding hyperparameters that yield the best result is a crucial step when developing deep learning solutions. This process called *hyperparameter optimization* is strictly related to

the goal to be achieved and the architecture being used. There are many ways to perform it, but the way I solved this problem was to use *Optuna*.

5.1.1 What is Optuna and how it works

Optuna is an automated hyperparameter optimization software framework that is knowingly invented for machine learning-based tasks and is framework-agnostic [16]. This means that you can use it with any machine learning or deep learning framework.
Optuna works by running a series of experiments to evaluate different combinations of hyperparameter values. Optuna uses a Bayesian optimization algorithm to guide the search process, which can help find the best hyperparameters more efficiently than other methods, such as grid search or random search.
The main advantages of Optuna are the following:

- efficiency: Optuna can find the best hyperparameters more efficiently than other methods, such as grid search or random search;
- ease of use: Optuna has a simple API that makes it easy to use without the need to modify your code;
- flexibility: Optuna can be used with any deep learning framework and can optimize any number of hyperparameters;
- distributed training: Optuna supports distributed training, which can make it faster to find the best hyperparameters for large models.

5.1.2 Basic structure

When we want to optimize a model, we must first define the hyperparameters that will be optimized by the process. This can be done using a built-in Optuna object that allows us to define the hyperparameter values called *Trial*. The Trial object represents a single experiment that Optuna is running. It contains different properties such as:

- the hyperparameter values that were used for the experiment;
- the objective function value for the experiment;
- the status of the experiment (running, finished, pruned).

Trial objects can be used to track the progress of the hyperparameter optimization process and to get insight into the performance of different hyperparameter values. Another important object of Optuna is the *Study* object that is a collection of trials. It represents a single hyperparameter optimization experiment. Studies can be used to optimize different deep learning models, different datasets, and different hyperparameter spaces.

```
# 1. Define the objective function
def objective(trial):
    # Define hyperparameters and model
    # Train and evaluate the model
    # Return the evaluation metric

# 2. Create a Study Object
study = optuna.create_study(direction='maximize')

# 3. Run the Optimization Process
study.optimize(objective, n_trials=100)
```

Listing 5.1: Optuna basic structure

The basic structure of an optimization process (listing 5.1) consists of the following steps:

1. defining the objective function: in this process an evaluation metric must be returned (for example test accuracy) to evaluate the trial with the defined hyperparameters;

2. creating a study object: create a study object that can be used to keep track of the experiments. A "direction" must be defined to tell Optuna to maximize the evaluation metric (accuracy for instance) or to minimize it (loss for instance);

3. running the optimization process: the optimization process is run for "n_trials" iterations, each time with different hyperparameters according to bayesian optimization search.

At the end of a study, the study object contains all the information about the optimization that was just performed. Some useful information that can be retrieved includes: the best combination of hyperparameters found, according to the evaluation metric, the combinations of 'top n' and 'worst n' hyperparameters, to provide insights into which combinations are good and which are not, and various graphs that can be used to visualize the importance of the hyperparameter, the optimization history and correlations between the values of the hyperparameters and the values of the objective function values.
All of this helped me design the best architecture for both the LSTM model (Section 5.2) and CGRNN model (Section 5.3).

5.2 Long Short-Term Memory Model (LSTM)

As stated in [25], LSTM models have not been used for seizure prediction, so there are no references in the literature on an optimal internal architecture. Therefore, I rely heavily on Optuna to find the best possible architecture.
The general structure can be seen in figure 5.1.

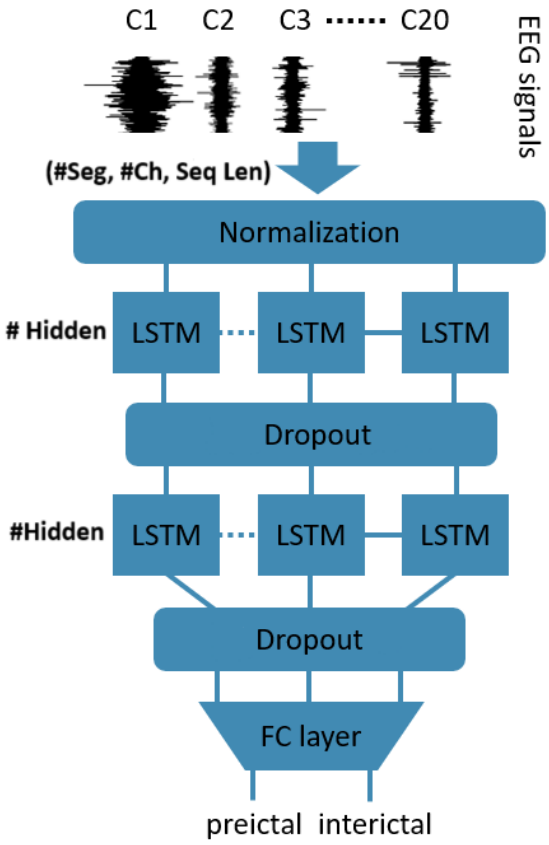


Figure 5.1: LSTM model inspired by [25]

The raw signals, normalized as stated in 4.2.2, flow in a two-layer LSTM structure where the temporal features are automatically extracted by the model. Between these two layers, there is a dropout layer that regularizes the model to rely on different sets of features to prevent overfitting.
Before being used to classify preictal and interictal data, these features pass through a dropout layer and then a fully connected layer to discriminate between the two classes.
This architecture is quite simple, but I found no information on LSTM layers, the number of neurons on each layer, dropout values, or many other parameters in the literature. To address this problem, I optimized both architectural and learning hyperparameters using Optuna.

Here are the hyperparameters that I tested:

- Architectural:
  - **number of LSTM layers**: 1, 2, 3;
  - **number of hidden neurons**: 32, 64, 128, 256.
- Learning:
  - **learning rate**: from 1e-5 up to 0.1;
  - **batch size**: 16, 32, 64, 128, 256;
  - **dropout value between LSTMs**: form 0.1 up to 0.9;
  - **dropout value before fc-layer**: form 0.1 up to 0.9;
  - **number of training epochs**: 50, 100, 150, 200.

The entire optimization process took 1 day and 22 hours to complete successfully, in 45 trials in which the evaluation metric was the **accuracy** on the **test** set with a 5-fold RCV procedure.
Figure 5.2 shows the relative importance of each hyperparameter in predicting the values of the objective function.

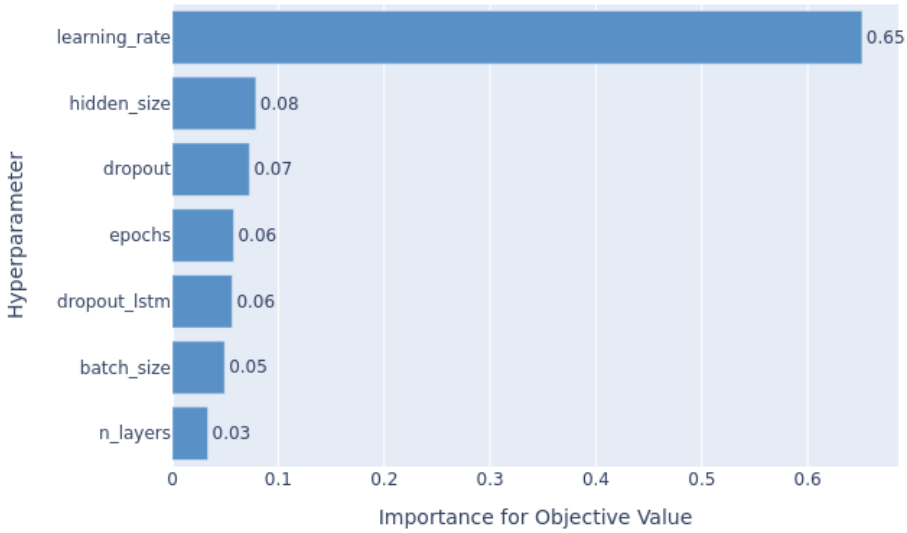


Figure 5.2: LSTM optimization: parameter importance

The most important hyperparameters that have the greatest impact on performance are the learning rate and the hidden size of each LSTM layer, followed by the two dropout values, the batch size, and the number of epochs required to properly train the model. The number of LSTM layers is the least important hyperparameter. This could be because the number of hidden units per layer has a greater impact on the model's capacity (ability to learn and represent complex patterns) than the number of layers, which can cause overfitting due to the increased computational complexity of the model.

In figure 5.3 there is the optimization history plot for the 45 trials. As Optuna explores different hyperparameter combinations, we can see that is capable of finding better and better solutions even if there is some randomness in the optimization process, as the objective function value can fluctuate from trial to trial.

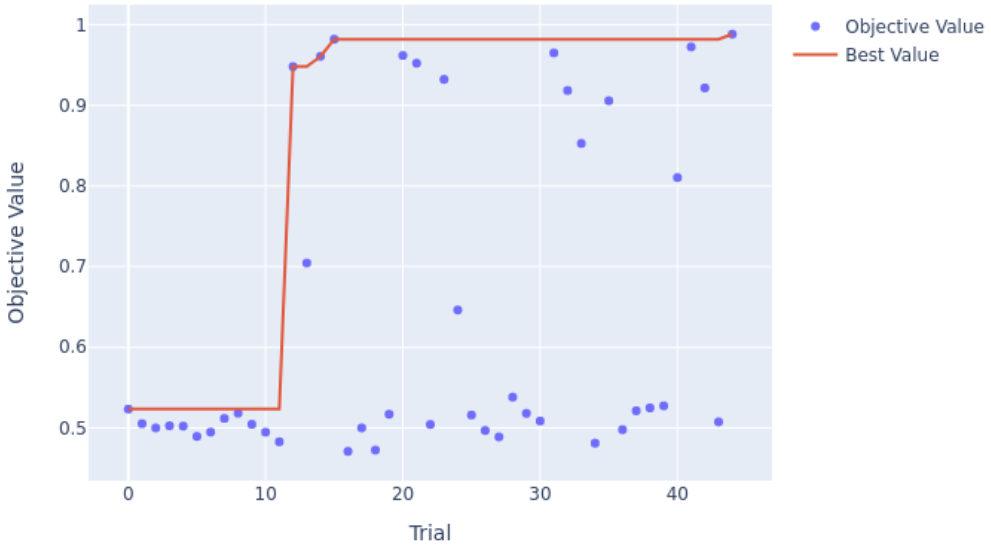


Figure 5.3: LSTM optimization: history plot

Overall, the optimization history plot shows that Optuna was able to find a good set of hyperparameters for the model that are the following:

- Architectural:
  - **number of LSTM layers**: 2;
  - **number of hidden neurons**: 32.
- Learning:
  - **learning rate**: 0.0048;
  - **batch size**: 32;
  - **dropout value between LSTMs**: 0.137;
  - **dropout value before fc-layer**: 0.3;
  - **number of training epochs**: 100.

This is the final LSTM architecture that was used to obtain the results presented in chapter 6.

5.3 Convolutional Gated RNN model (CGRNN)

Considering that the input is an EEG, an obvious approach would be stacking multiple LSTMs as the previous model in figure 5.1 but I decided to use GRUs

since they use fewer parameters than LSTMs and hence offer a faster training time while requiring data to generalize.
GRUs, such as LSTMs, are architectures for handling sequential input data, which has led to state-of-the-art accuracy in various pattern recognition tasks. However, when applied to relatively long input time series data, this approach turns out to be computationally very intensive and time-consuming to train.
To solve this problem, S. Roy et al. [20], tried to downsample the data to an acceptable size before feeding them into the GRUs. To mitigate these problems, they used multiple 1D convolution layers (Conv1D) with strides larger than 1, allowing the network to learn to appropriately reduce the input signal automatically.

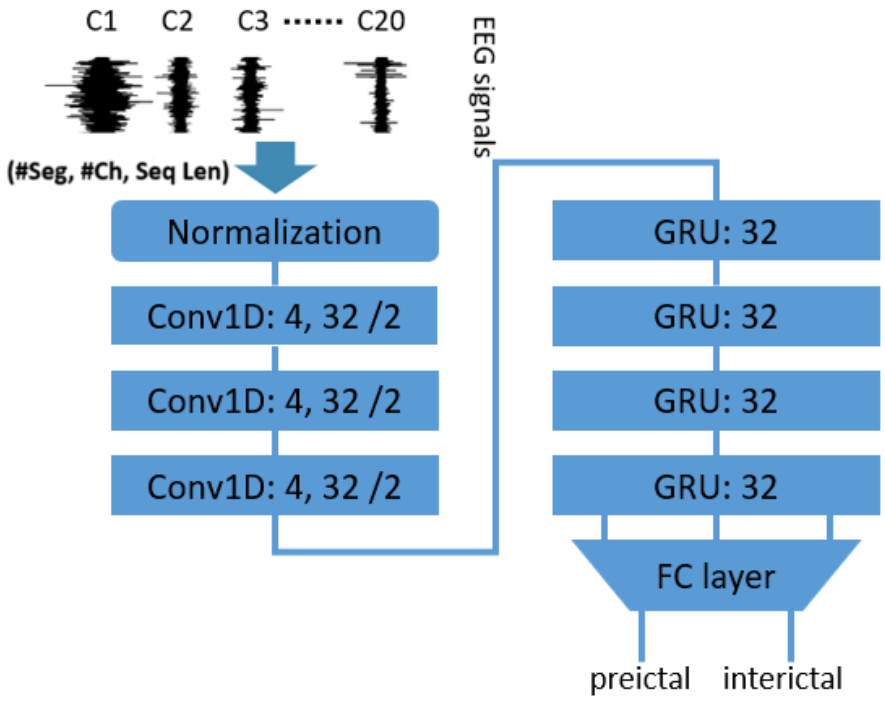


Figure 5.4: CGRNN model inspired by [20]

The resulting architecture (CGRNN in figure 5.4) is a combination of Conv1D layers followed by stacked GRU layers.
The main advantages of Conv1D layers are two:

1. learn to downsample the input signal, reducing the shapes of the input as we move towards deeper layers. This is important for GRU layers, which are the most computationally expensive part of the network to train;

2. learn local temporal dependencies from neighboring data points in the EEG segments.

The GRU layers are then fed with the features extracted by the Conv1D layers, which are responsible for capturing both short- and long-term dependencies in the data.

Unlike LSTM in section 5.2 the paper [20] clearly described the architecture, so Optuna was only used to find the best learning rate, batch size, and epochs, which took 13 hours and 30 minutes over 30 trials.

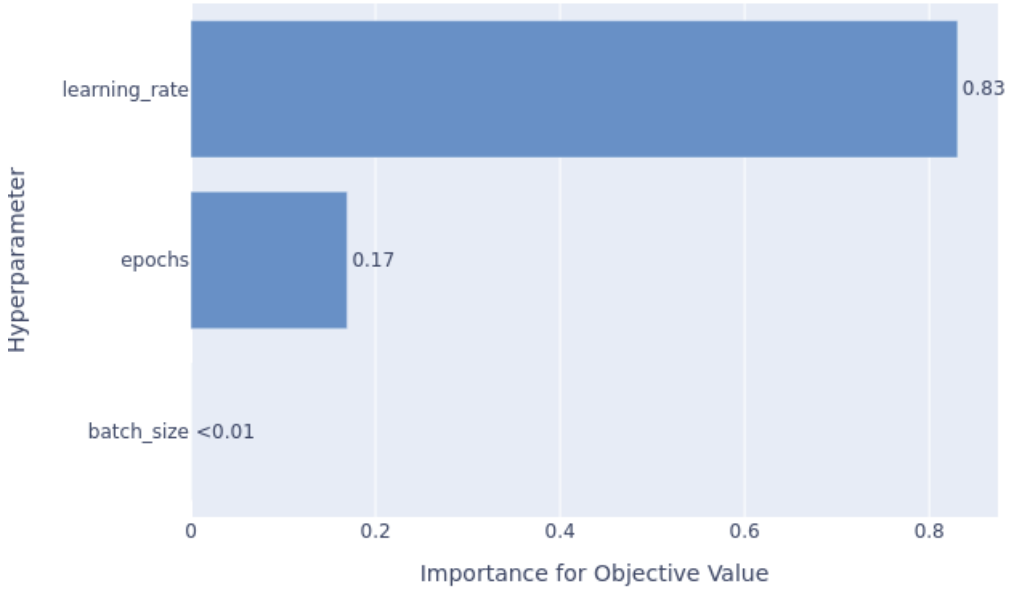


Figure 5.5: CGRNN optimization: hyperparameter importance

Figure 5.5 shows the hyperparameter importance. As with the LSTM model, the learning rate is the most important hyperparameter, while batch size plays a marginal role in the learning process. This is likely because the architecture is already optimized in the paper, making it less sensitive to changes in batch size than to changes in learning rate.
Figure 5.6 there is the optimization history plot for the 30 trials. The plot shows that the optimizer was able to find a good solution relatively quickly and that the objective value improved steadily over time, suggesting that the optimizer had found a near-optimal solution.

The final architecture of the CGRNN, that was used to get the results presented in chapter 6, is the following:

- Architectural:
  - **number of Conv1D layers**: 3 (from paper);
  - **kernel size for each Conv1D layer**: 4 (from paper);
  - **number of hidden neurons (Conv1D)**: 32 (from paper);
  - **stride for each Conv1D layer**: 2 (from paper);
  - **number of GRU layers**: 4 (from paper);
  - **number of hidden neurons (GRU)**: 32 (from paper).
- Learning:

– **learning rate**: 0.0017;
– **batch size**: 128;
– **number of training epochs**: 200.

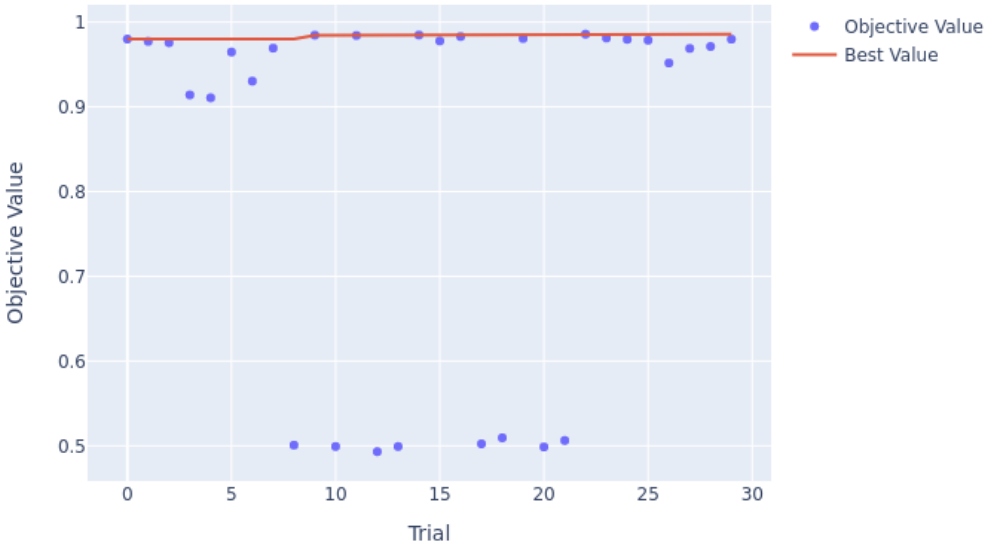


Figure 5.6: CGRNN optimization: history plot

6 | Results

In this chapter, I will present all the results I have been able to obtain with the four selected datasets. The various datasets are evaluated based on seven experiments: 5-fold cross validation, 5-fold cross validation with randomized channels, leave-one-patient-out, leave-one-patient-out with randomized channels, variance feature selection, variance difference feature selection, and wrapper feature selection

6.1 Introduction

This chapter summarizes all the results I have been able to obtain for the two models (LSTM and CGRNN). For each model, seven experiments were conducted with four different datasets for a total of 56 trials.
The first goal of these experiments was to evaluate the actual ability of the models to extract relevant temporal features for the prediction of epileptic seizures. The second goal was to evaluate which datasets contained the most relevant information so that future experiments could be carried out more accurately. This is particularly true for feature selection experiments.
As we have already mentioned in chapter 4, feature selection is the process of selecting a subset of the most relevant features from a dataset for a given task. This has many advantages from a computational perspective, but it also has advantages from a medical perspective.
Reducing the number of channels (which in our case are the features of the dataset) not only leads to improved model performance, but also to a reduction in the costs and time of electroencephalographic examinations, as it is possible to record fewer areas of the brain than those used in these experiments (Table 4.1).
Another advantage is that with careful selection of channels, it is possible to identify which areas of the brain are more or less responsible for which types of epileptic seizures, thus helping to develop a more accurate understanding of the origins and causes of this disease.

This chapter will be divided into datasets, and each dataset presents sections divided by experiments and models. This allows for a comparison of the models' performance for any given experiment.
To have a fair comparison between the two models, I decided to train even the

CGRNN for 100 epochs (instead of 200) since, as seen in the chapter 5 section 5.3, it has a nearly marginal impact on the learning capabilities of the model, and this is also confirmed in the many attempts that I have done.

All line graphs were created using the mean values' results from the 5 folds both for training and test. The cold colors (blue and green) are associated with training, while the warm colors (yellow and orange) are associated with testing. The values used to create the graphs are those obtained from the mean of the performances in the five folds.

6.2 Dataset: 5sec segments - 128Hz - no overlap

The dataset in table 6.1 is made up of 57.612 segments, each described by 20 channels, with a length of 640 for a final shape of (57.612, 20, 640). This comes from raw EEG signals that are subsampled at 128 Hz without overlap between segments.

Frequency	**Overlap**	**#Segments**	**#Channels**	**Seq.Len**
128	0	57.612	20	640

Table 6.1: Dataset: 5 second segments with no overlap and 128Hz frequency

6.2.1 5-Fold RCV

Metrics	**LSTM**	**CGRNN**	**p-value**
ACC	98.09 ± 0.28	96.56 ± 1.54	0.09
SEN	98.74 ± 0.73	98.89 ± 0.9	0.08
SPE	97.42 ± 0.92	96.16 ± 2.37	0.54

Table 6.2: Dataset 5s 128Hz 0s: summary result of the RCV procedure on the test set

In table 6.2 and figure 6.1 we find the results of the 5-fold cross-validation. As we can see, both models are able to achieve good results that exceed the 95% threshold for each proposed metric.
The LSTM model obtains more accurate and precise results with much lower variability than the CGRNN. However, the LSTM takes about 44 seconds per training epoch, compared to 15 seconds for the CGRNN.
In total, the LSTM takes 73 minutes to complete one training fold, compared to 25 minutes for the CGRNN.

The p-value is less than the threshold of 0.05 for all metrics reported. This suggests that the variability in the results is not due to the test that was performed, but to the intrinsic randomness of the training methods of the models. Both models are capable, for the RCV, of learning robust temporal features to predict an epileptic seizure within 30 minutes from onset.

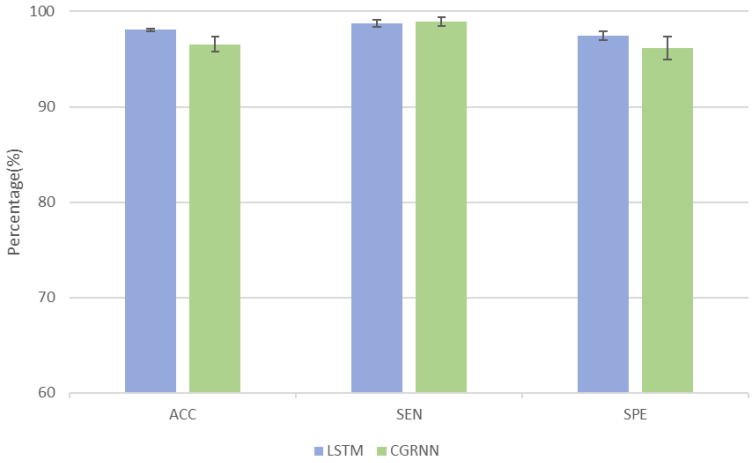


Figure 6.1: 5F RCV: comparison between LSTM and CGRNN on dataset 5s 128Hz 0 overlap. The intervals on the histograms (black lines) are the standard error for the corresponding metric. The data are the one in table 6.2

In figures 6.2 and 6.3 we can see a comparison of training and test procedures for the two models.
In these figures, we can see how both models are able to learn the temporal features without overfitting, but the main difference is that the CGRNN presents more linear curves with fewer sharp changes, especially in the test loss, suggesting a more stable and robust training phase.

6.2.2 5-Fold RCV randomized channels

Metrics	LSTM	CGRNN	p-value
ACC	98.13 ± 0.38	97.46 ± 0.66	0.13
SEN	98.11 ± 0.95	96.76 ± 1.8	0.23
SPE	98.15 ± 1.29	98.17 ± 0.53	0.98

Table 6.3: Dataset 5s 128Hz 0s: summary result of the RCV procedure on the test set with randomized channels

In table 6.3 and figure 6.4 can see the results for the RCV with randomized channels.
The p-value suggests that the two models behave in the same way, thus having no significant differences in terms of performance.

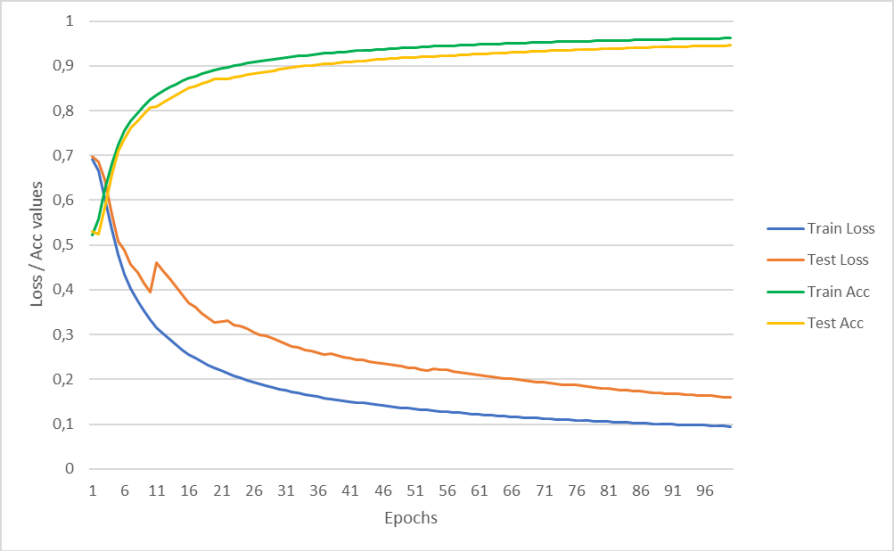


Figure 6.2: 5F RCV: training and test accuracy/loss for the LSTM model on dataset 5s 128Hz 0 overlap

In this case, the CGRNN presents smaller oscillations than in the previous case, suggesting that the oscillations are due to the random nature of the training.

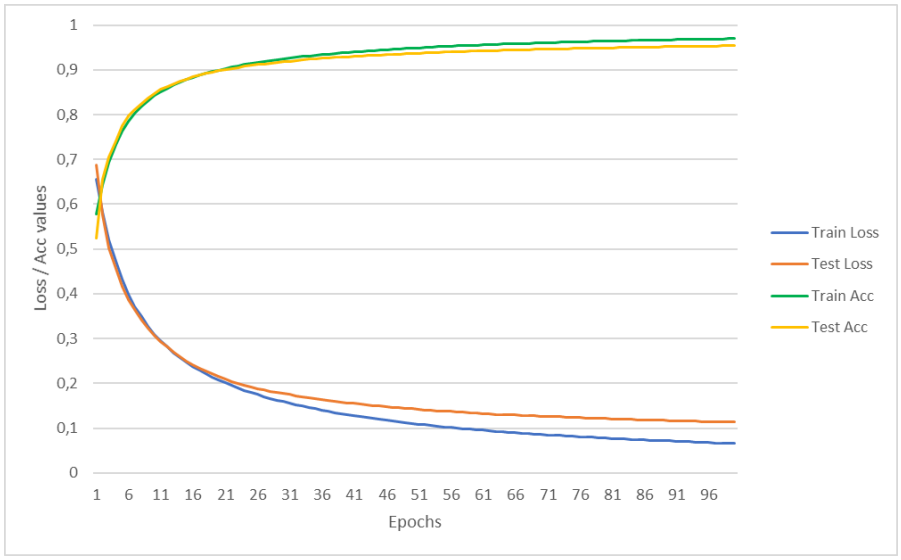


Figure 6.3: 5F RCV: training and test accuracy/loss for the CGRNN model on dataset 5s 128Hz 0 overlap

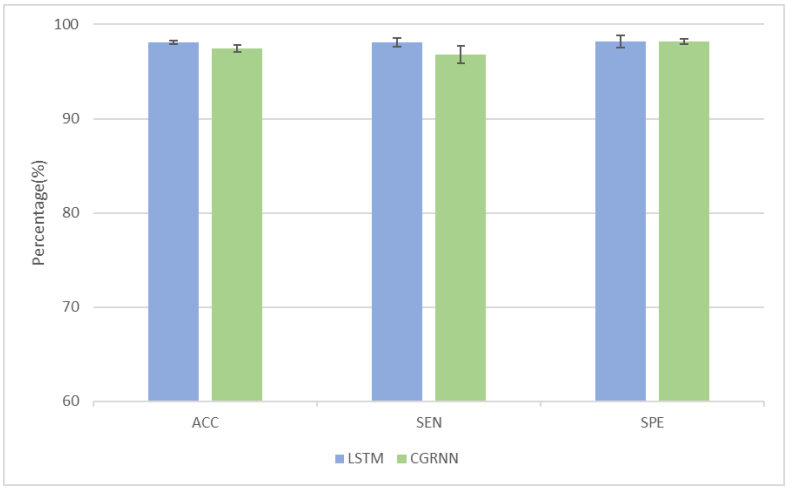


Figure 6.4: 5F RCV: comparison between LSTM and CGRNN on dataset 5s 128Hz 0 overlap with randomized channels. The data is the one in table 6.3

Figures 6.5 and 6.6 show the loss and accuracy for both training and test. Even in this case the CGRNN 6.6 presents smoother curves than the one for LSTM 6.5 that, as in the previous case, presents some spikes in the loss values.

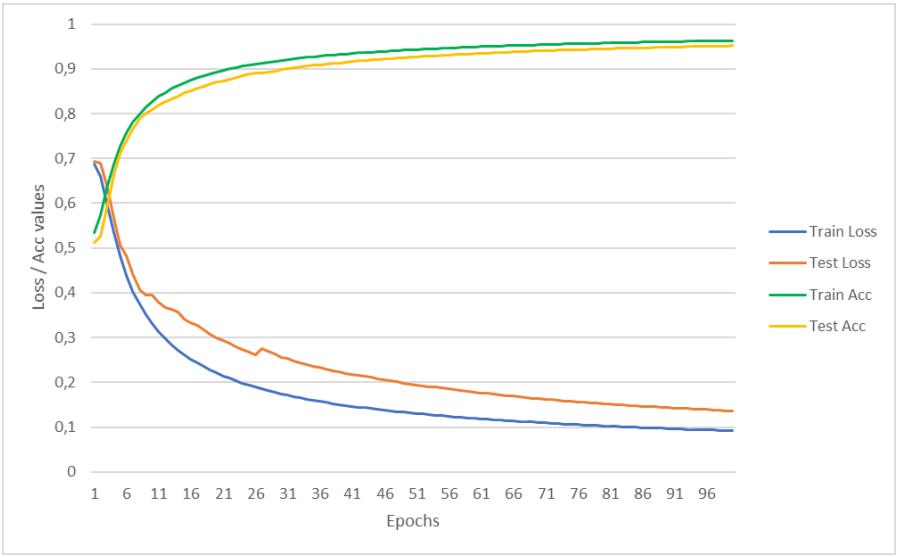


Figure 6.5: 5F RCV: training and test accuracy/loss for the LSTM model on dataset 5s 128Hz 0 overlap with randomized channels

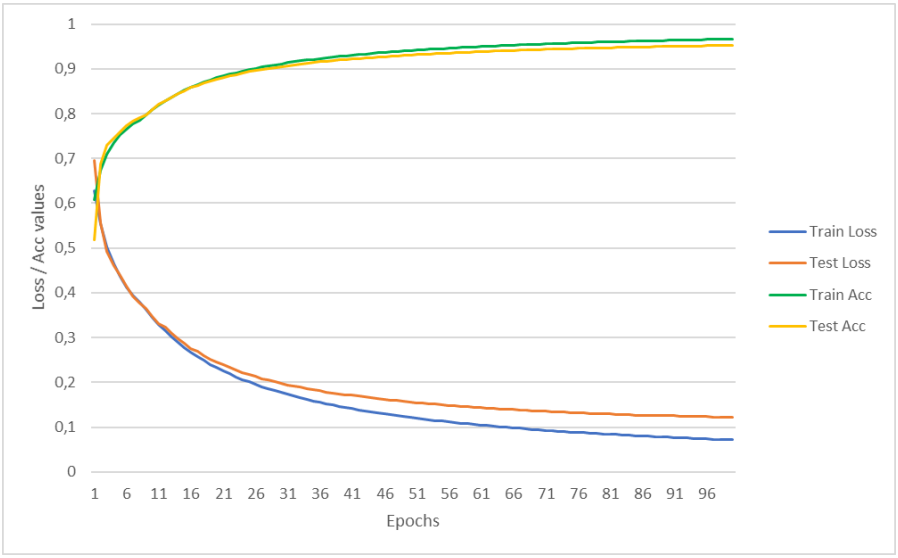


Figure 6.6: 5F RCV: training and test accuracy/loss for the CGRNN model on dataset 5s 128Hz 0 overlap with randomized channels

6.2.3 LOO

Metrics	LSTM	CGRNN	p-value
ACC	62.94 ± 8.71	56.88 ± 6.5	0.28
SEN	64.63 ± 12.51	55.33 ± 13.3	0.32
SPE	61.12 ± 11.21	58.54 ± 11.42	0.75

Table 6.4: Dataset 5s 128Hz 0s: summary result of the LOO procedure on the test set

Table 6.4 and figure 6.7 show the results for the LOO validation procedure on the given dataset. As we stated in the definition of the problem in chapter 4, the LOO procedure evaluates the ability of the model to predict seizures of unknown patients. This task is quite challenging, and this is shown in the tables where the results, compared to the RCV procedure, get a huge drop.
The results suggest that the LSTM model has a slight advantage over the CGRNN model in terms of accuracy and sensitivity. However, the difference between the two models is not statistically significant (the p-values are higher than the threshold at 5%).
In this experiment, both LSTM and CGRNN have huge confidence intervals in all metrics caused by uncertain results. This was expected due to the challenging nature of the task, especially on a dataset with high variance of data (different patients, different types of epilepsy, and different amounts of data recorded).
Table 6.5 and table 6.6 show in detail the results for each patient.
It is difficult to draw any firm conclusions about the overall performance of the models from these results, as performance varies significantly from patient to patient. However, there are some general trends that can be observed. There are some patients like Patient 6 or Patient 13 who, in both models, have good results, which means that the models were able to detect their seizures even if they were an 'unknown' patient. On the other hand, there are patients like Patient 27 and Patient 28 who perform very poorly.
The fact that some patients have good performance and others do not can be due to many reasons, such as

- some patients in the training set present a type of seizure that is similar to one of the left out patients (good performances);
- some patients have a rare type of seizure and they are the only one in the dataset with such disease (poor performances);
- seizures can originate from different areas of the brain, for which there may be many or few data available. This can lead to good or poor performance accordingly;
- it is possible that patient data may have been recorded under optimal conditions and have good signal quality, resulting in good performance, or, on the contrary, may have many artifacts, resulting in poorer performance.

There are many things that must be considered, in addition to the fact that LOO is a very complex task and should be carefully analyzed even with the help of specialized clinicians. The fact that it is possible to differentiate patients for whom the procedure works well from others for whom it does not work well is a starting point that deserves further exploration.

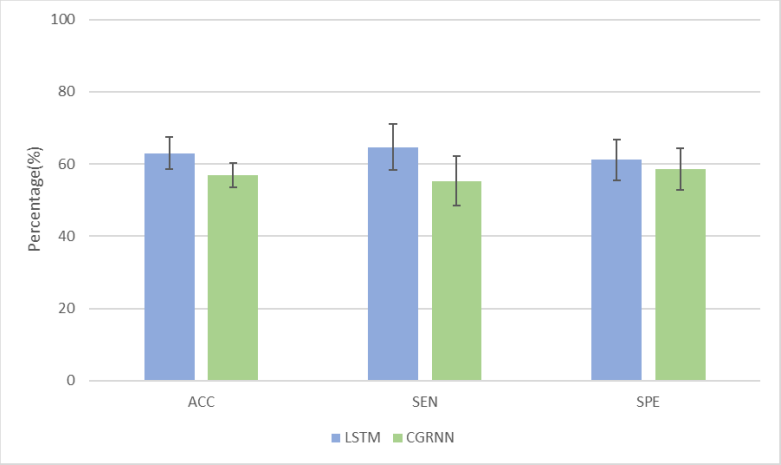


Figure 6.7: LOO: comparison between LSTM and CGRNN on dataset 5s 128Hz 0 overlap. The data are the one in table 6.4

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	719	2681	18,39	12,66	24,38
Patient 2	360	5275	60,51	97,78	21,51
Patient 3	720	5431	92,78	94,72	90,83
Patient 4	1440	9195	41,60	6,25	76,94
Patient 5	360	1738	79,12	68,33	90,41
Patient 6	2763	17064	92,73	98,19	87,23
Patient 7	360	1728	32,24	37,50	26,74
Patient 8	2160	11072	72,62	93,19	52,04
Patient 9	720	2765	77,99	91,25	64,72
Patient 10	462	3006	87,17	93,72	80,18
Patient 11	720	3595	96,67	95,69	97,64
Patient 12	1080	14235	50,14	98,70	0,85
Patient 13	720	2318	98,06	100,00	96,11
Patient 14	540	1987	78,22	85,19	70,93
Patient 15	810	3606	65,25	55,06	75,70
Patient 16	468	3744	56,36	62,61	50,00
Patient 17	720	3296	75,49	54,17	96,81
Patient 18	2427	18868	56,31	64,98	47,57
Patient 19	1080	5610	49,81	2,13	98,21
Patient 20	720	6516	48,12	50,14	46,11
Patient 21	1800	16731	81,45	99,89	62,84
Patient 22	1440	6989	72,43	60,28	84,58
Patient 23	835	6306	76,62	98,80	54,28
Patient 24	590	3930	48,87	93,90	1,60
Patient 25	720	3196	71,53	79,31	63,75
Patient 26	1080	8632	73,09	48,06	98,50
Patient 27	720	5630	3,26	3,47	3,06
Patient 28	1080	7916	27,29	18,43	36,28
Patient 29	1192	3661	41,30	10,15	72,87

Table 6.5: Dataset 5s 128Hz 0s test set: results for the LOO procedure with LSTM divided by patient.

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	719	2681	48,93	0,00	100,00
Patient 2	360	5275	64,84	99,17	20,71
Patient 3	720	5431	65,77	90,56	39,83
Patient 4	1440	9195	40,02	12,78	68,53
Patient 5	360	1738	51,41	19,72	92,14
Patient 6	2763	17064	86,90	78,32	95,55
Patient 7	360	1728	46,56	53,06	38,21
Patient 8	2160	11072	49,50	59,31	39,24
Patient 9	720	2765	67,05	42,08	93,17
Patient 10	462	3006	78,24	91,77	63,82
Patient 11	720	3595	48,86	0,00	100,00
Patient 12	1080	14235	52,73	98,15	2,07
Patient 13	720	2318	82,10	99,44	63,95
Patient 14	540	1987	32,32	15,19	51,45
Patient 15	810	3606	78,84	94,44	61,43
Patient 16	468	3744	31,25	26,92	35,98
Patient 17	720	3296	80,82	70,56	91,57
Patient 18	2427	18868	62,42	63,12	61,67
Patient 19	1080	5610	48,83	13,70	88,02
Patient 20	720	6516	71,38	50,00	93,75
Patient 21	1800	16731	80,58	98,22	62,78
Patient 22	1440	6989	64,28	93,68	33,50
Patient 23	835	6306	55,41	99,28	11,22
Patient 24	590	3930	50,78	95,59	3,74
Patient 25	720	3196	55,40	58,89	51,74
Patient 26	1080	8632	64,94	46,57	85,43
Patient 27	720	5630	24,93	4,58	46,22
Patient 28	1080	7916	16,41	24,63	7,23
Patient 29	1192	3661	48,22	4,78	94,78

Table 6.6: Dataset 5s 128Hz 0s test set: results for the LOO procedure with CGRNN divided by patient

6.2.4 LOO randomized channels

Metrics	LSTM	CGRNN	p-value
ACC	58.24 ± 8.57	53.70 ± 6.08	0.43
SEN	60.49 ± 13.53	48.37 ± 12.2	0.2
SPE	56.05 ± 12.36	59.31 ± 11.28	0.7

Table 6.7: Dataset 5s 128Hz 0s: summary result of the LOO procedure on the test set with randomized channels

In the case of randomized channels, the same consideration holds. Table 6.7 and Figure 6.8 suggest that LSTM has a small advantage over CGRNN in accuracy and specificity, while it is slightly worse in sensitivity. Overall, the two models perform similarly even in the case of randomized channels because the p-values, for all the three metrics, are grater than the 5% threshold.
Even in this case the confidence intervals are large, suggesting uncertainty in the degree of prediction with values similar to the one presented in table 6.4.
Tables 6.8 and 6.9 show the results for each patient. As before, we have patients in which the models perform well (Patient 11) and others in which they perform very poorly (Patient 27).

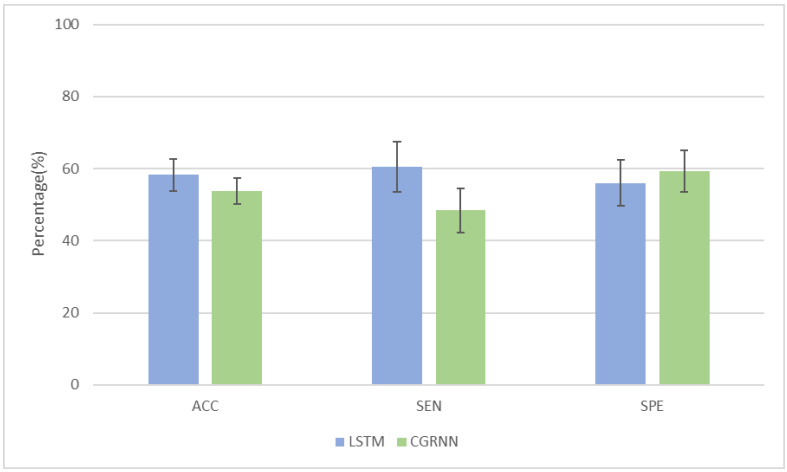


Figure 6.8: LOO with randomized channels: comparison between LSTM and CGRNN on dataset 5s 128Hz 0 overlap. The data are the one in table 6.7

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	719	2681	26,99	19,47	34,83
Patient 2	360	5275	62,64	100,00	23,55
Patient 3	720	5431	57,50	83,75	31,25
Patient 4	1440	9195	32,43	57,99	6,88
Patient 5	360	1738	68,75	43,06	95,64
Patient 6	2763	17064	96,18	96,85	95,51
Patient 7	360	1728	49,29	3,61	97,09
Patient 8	2160	11072	65,93	82,78	49,07
Patient 9	720	2765	69,79	95,97	43,61
Patient 10	462	3006	77,23	95,89	57,37
Patient 11	720	3595	96,60	97,22	95,97
Patient 12	1080	14235	53,03	99,91	5,45
Patient 13	720	2318	94,31	98,47	90,14
Patient 14	540	1987	60,80	24,26	99,03
Patient 15	810	3606	74,31	56,17	92,91
Patient 16	468	3744	7,87	4,49	11,30
Patient 17	720	3296	57,64	15,42	99,86
Patient 18	2427	18868	57,51	60,73	54,26
Patient 19	1080	5610	49,35	1,85	97,56
Patient 20	720	6516	39,38	50,14	28,61
Patient 21	1800	16731	87,30	98,94	75,56
Patient 22	1440	6989	58,92	94,03	23,82
Patient 23	835	6306	67,91	98,68	36,91
Patient 24	590	3930	68,14	83,05	52,49
Patient 25	720	3196	59,24	93,06	25,42
Patient 26	1080	8632	74,44	59,72	89,38
Patient 27	720	5630	3,26	0,69	5,83
Patient 28	1080	7916	22,57	19,17	26,03
Patient 29	1192	3661	49,37	18,88	80,27

Table 6.8: Dataset 5s 128Hz 0s test set with randomized channels: results for the LOO procedure with LSTM divided by patient

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	719	2681	27,34	48,12	5,66
Patient 2	360	5275	61,56	98,33	14,29
Patient 3	720	5431	82,67	88,75	76,31
Patient 4	1440	9195	42,97	14,51	72,75
Patient 5	360	1738	47,81	15,28	89,64
Patient 6	2763	17064	68,77	55,81	81,83
Patient 7	360	1728	38,44	40,28	36,07
Patient 8	2160	11072	62,76	74,58	50,39
Patient 9	720	2765	71,80	46,67	98,11
Patient 10	462	3006	56,14	90,48	19,59
Patient 11	720	3595	94,96	93,06	96,95
Patient 12	1080	14235	51,81	96,11	2,38
Patient 13	720	2318	48,86	0,00	100,00
Patient 14	540	1987	35,64	35,74	35,54
Patient 15	810	3606	47,27	0,00	100,00
Patient 16	468	3744	26,90	15,81	39,02
Patient 17	720	3296	63,21	42,64	84,74
Patient 18	2427	18868	54,81	44,79	65,35
Patient 19	1080	5610	46,92	17,96	79,24
Patient 20	720	6516	68,82	58,89	79,22
Patient 21	1800	16731	49,78	0,00	100,00
Patient 22	1440	6989	74,18	90,69	56,90
Patient 23	835	6306	71,33	98,08	44,39
Patient 24	590	3930	59,38	65,76	52,67
Patient 25	720	3196	61,65	65,42	57,70
Patient 26	1080	8632	76,51	63,80	90,70
Patient 27	720	5630	11,93	3,89	20,35
Patient 28	1080	7916	16,02	11,11	21,49
Patient 29	1192	3661	37,28	26,43	48,92

Table 6.9: Dataset 5s 128Hz 0s test set with randomized channels: results for the LOO procedure with CGRNN divided by patient

6.2.5 Filter feature selection

Method	Ranked channels (desc)
Variance	Pz-C4-P4-P3-F7-Fp2-Cz-F3-T7-O1-T8-F8-Fp1-Fz-Oz-C3-P8-F4-O2-P7
Variance-diff	Fz-C4-C3-F4-P3-Cz-P7-O2-Fp1-Fp2-P4-Oz-F7-F8-O1-P8-F3-Pz-T7-T8

Table 6.10: Channels in descending order ranked by variance and variance-difference filter methods

In table 6.10, the EEG channels are listed in descending order of importance, calculated using the variance feature selection and variance difference feature selection methods.
To evaluate the actual informativeness of the channels and therefore the effectiveness of the feature selection methods, the two models were retrained five times, each using a different number of channels. The evaluation method used is 5-fold RCV. The channels were taken in descending order as reported in Table 6.10. The experiments carried out involve the use of one, three, five, ten, and fifteen channels. In order to compare the CGRNN fairly with the wrapper methods, the latter was also trained with sixteen channels. The results of the variance-based filter method are reported in Table 6.11 and in Figure 6.9.
With a single channel (Pz), both models obtained relatively good results of about

Variance		ACC(%)	SEN(%)	SPE(%)
1 channel	**LSTM**	71,70	69,48	73,91
	CGRNN	71,53	69,71	73,38
3 channels	**LSTM**	79,95	64,72	95,45
	CGRNN	82,34	83,26	81,42
5 channels	**LSTM**	89,53	83,41	95,81
	CGRNN	86,15	88,53	83,78
10 channels	**LSTM**	91,96	86,45	97,44
	CGRNN	92,85	92,29	93,40
15 channels	**LSTM**	95,59	92,99	98,27
	CGRNN	96,20	95,17	97,22
16 channels	**LSTM**	—	—	—
	CGRNN	95,63	95,23	96,02

Table 6.11: Dataset 5s 128Hz 0s: results for the feature selection procedure using the variance filter method.

70% in all three metrics. As expected, with increasing number of channels, the performance of the models improves significantly.
Moving from one channel to three (Pz-C4-P4), the performance of the CGRNN

increased dramatically, surpassing the LSTM for accuracy and sensitivity, but not for specificity.
The trend is the same: the more channels are used for training, the better the performance of the models. It appears that all channels are necessary for predicting an epileptic seizure, and that eliminating any of them would lead to a slight, yet noticeable, decrease in performance.
One thing to note for the CGRNN is that the results with sixteen channels (Pz-C4-P4-P3-F7-Fp2-Cz-F3-T7-O1-T8-F8-Fp1-Fz-Oz-C3) show a slight decrease in performance compared to the results recorded with fifteen channels (Pz-C4-P4-P3-F7-Fp2-Cz-F3-T7-O1-T8-F8-Fp1-Fz-Oz). This may be due to the randomness of the training method because, as reported in Table 6.2 where the 5-fold RCV was performed on all channels, the results are similar to those reported with 15 or 16 channels. This may be an indication that only the last four or five channels (C3-P8-F4-O2-P7) contain information that is not very relevant for the experiments conducted in this paper.

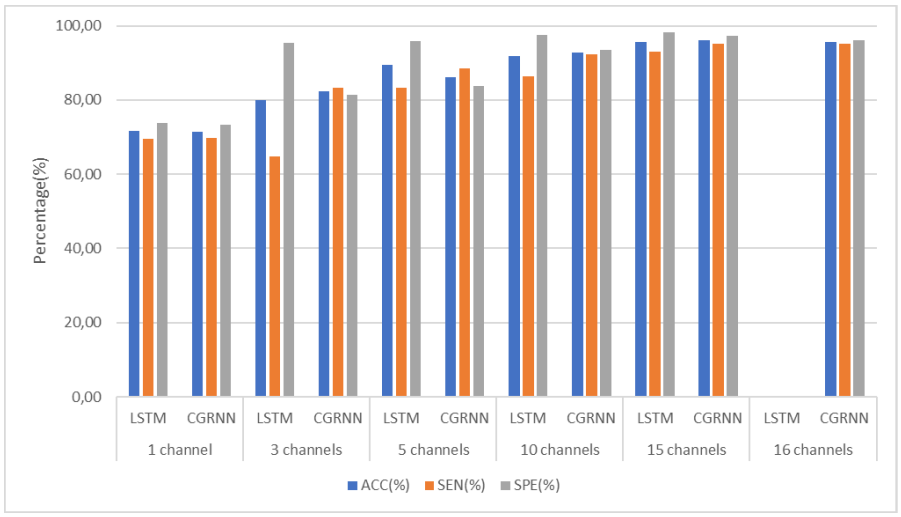


Figure 6.9: Dataset 5s 128Hz 0s: result for the variance feature selection based on the results on table 6.11

Even if we use the channels in order of importance according to the variance difference, the considerations made before are still valid. These can be seen in table 6.12 and figure 6.10.
In this case, the use of a single channel (Fz) results in the CGRNN achieving similar performances to those in the case of variance. The LSTM, on the other hand, results in poor performances. With the use of three channels (Fz-C4-C3), the performance of both models increases drastically as in the previous case.
This behavior suggests that, even if the two filtering methods give different scores to the channels, all of them contribute to the predictions of epileptic seizures, regardless of the order in which they are used.
Here are some considerations. In the case of the variance filtering method, one of the channels with a lower score is F4, which in the case of the variance difference method is one of the channels with a higher score. This means that F4 is never used in the

Variance difference		ACC(%)	SEN(%)	SPE(%)
1 channel	**LSTM**	54,46	70,16	38,88
	CGRNN	71,79	69,02	74,52
3 channels	**LSTM**	81,68	72,31	91,63
	CGRNN	84,55	82,16	87,01
5 channels	**LSTM**	87,90	80,22	95,48
	CGRNN	88,44	87,15	89,74
10 channels	**LSTM**	92,43	85,91	98,96
	CGRNN	92,88	92,35	93,43
15 channels	**LSTM**	95,68	92,34	99,15
	CGRNN	95,10	96,94	93,28
16 channels	**LSTM**	—	—	—
	CGRNN	96,06	96,47	95,64

Table 6.12: Dataset 5s 128Hz 0s: results for the feature selection procedure using the variance difference filter method.

variance filtering method (because it is the 18th channel, for which no experiments have been conducted), while it is used almost always in the variance difference method (because it is the fourth channel in order of importance). Whether used or not, the models still achieve good performance. This suggests that the channels contain redundant information that can be retrieved from other channels if it is not available from a specific channel. Furthermore, the total information contained in each channel is different enough to lead to optimal performance when all channels are used. This is demonstrated by the fact that performance improves as the number of channels increases, even if the improvement becomes less significant with the addition of more channels.

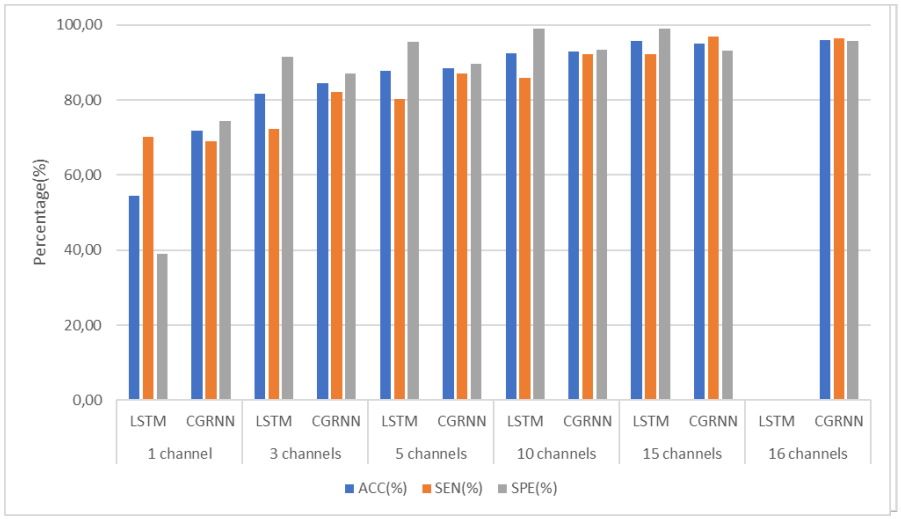


Figure 6.10: Dataset 5s 128Hz 0s: result for the variance difference feature selection based on the results on table 6.12

6.2.6 Wrapper feature selection

Model	#channels	Selected channels
LSTM	15	Cz-F4-F7-F8-Fp1-Fp2-Fz-O1-O2-Oz-P4-P8-Pz-T7-T8
CGRNN	16	C3-Cz-F3-F8-Fp1-Fp2-Fz-O1-O2-Oz-P4-P7-P8-Pz-T7-T8

Table 6.13: Channels selected by the wrapper feature selection method presented in listing 4.1

Table 6.13 shows the channels selected by the procedure in listing 4.1. This procedure reduced the number of channels from twenty to fifteen for LSTM and sixteen for CGRNN, without degrading the initial performance (calculated using 5-fold RCV with twenty channels).
The wrapper method applied to LSTM eliminated channels F3, C3, C4, P3, and P7, while applying it to CGRNN, the eliminated channels are C4, F4, F7, and P3.
As shown in table 6.14 and figure 6.11, the difference in channels does not lead to

Metrics	LSTM	CGRNN	p-value
ACC	97.26 ± 0.21	96.87 ± 0.43	0.16
SEN	97.17 ± 0.67	96.95 ± 0.96	0.73
SPE	97.33 ± 0.95	96.77 ± 0.43	0.31

Table 6.14: Dataset 5s 128Hz 0s: summary result of the wrapper feature selection method presented in listing 4.1

a significant difference in performance in the two models as discussed for the filter methods. Both models can achieve excellent results, exceeding 96% in each metric,

with relatively low variability, less than 1. This indicates that the proposed results are fairly accurate.
A significant statistic is the p-value that, for each metric, is greater than the threshold value set at 5%. This tells us that the results, for each model, depend on random fluctuations in the learning method and, therefore, the fact that the wrapper method selected fifteen channels for the LSTM and sixteen for the CGRNN does not influence the actual performance capabilities of the latter. This leads us to say, with a certain degree of confidence, that the CGRNN even with fifteen channels would have reached a similar performance to those obtained with sixteen channels.
The main reason why CGRNN was unable to further reduce the number of channels is due to the wrapper feature selection stopping methodology. This methodology does not round out the results, which means that even a small fluctuation of 0.01 can result in a performance downgrade. This leads to stopping the channel selection.
In table 6.15 and figure 6.12, we can see a comparison between the results obtained

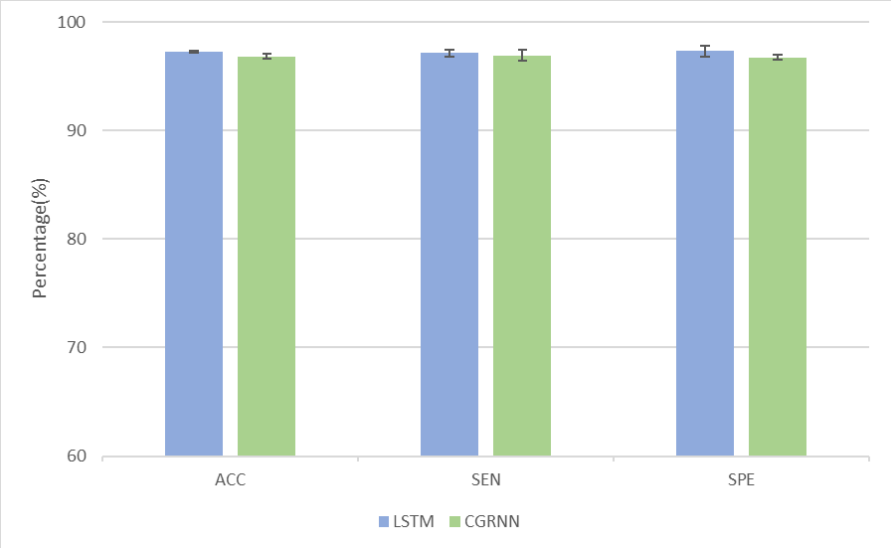


Figure 6.11: Dataset 5s 128Hz 0s: result for the wrapper feature selection method on the results on table 6.14

by the three feature selection methods on the LSTM model using fifteen channels (in order to allow comparison with the wrapper method).
As already discussed in chapter 4, the wrapper method results in slightly better performance than the other two methods, at the cost of a higher computational and time expense.
In table 6.16, we can see that the p-values associated with accuracy are all well above the 5% threshold. This is an indication that the three feature selection methods, even though they select different channels, allow the model to learn relevant temporal features and achieve good performance in the 5-fold cross-validation evaluation method. Despite the different feature selection methods, these p-values suggest what was already discussed previously: all channels contain relevant and redundant information that the model can exploit to predict epileptic seizures even when some channels are completely removed from the training phase. All three

Method	ACC(%)	SEN(%)	SPE(%)
Variance	95,59	98,27	92,99
Var-Diff	95,68	99,15	92,34
Wrapper	97,60	96,92	98,28

Table 6.15: Dataset 5s 128Hz 0s: comparison between the result of the three feature selection methods on LSTM with 15 channels

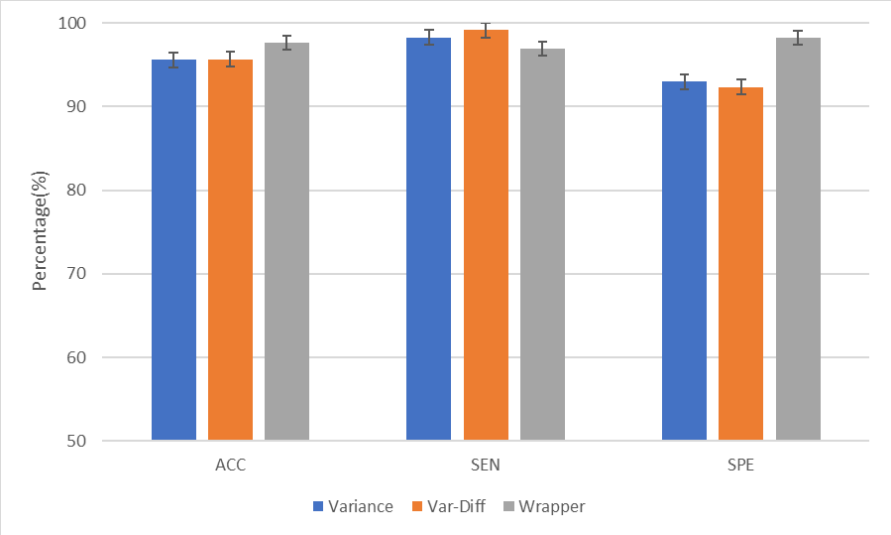


Figure 6.12: Dataset 5s 128Hz 0s: comparison between the three feature selection methods on LSTM using the results in table 6.15

methods select a good subset of channels that the model can exploit effectively. In table 6.17 and figure 6.13, we can see a comparison between the results obtained by the three feature selection methods on the CGRNN model using sixteen channels (in order to allow comparison with the wrapper method).
In this case, as in the case of LSTMs, the wrapper method results in slightly better performance than the other two methods in exchange for a higher computational cost.

In table 6.18, we can find the p-values, calculated on accuracy, that compare the three feature selection methods applied to the CGRNN.
In this case, as in the LSTM, the p-values indicate that the three methods, among

Method	p-value
Wrapper-Variance	0,59
Wrapper-VarDiff	0,95
Variance-VarDiff	0,56

Table 6.16: Dataset 5s 128Hz 0s: p-value between the various feature selection methods for LSTM

Method	ACC(%)	SEN(%)	SPE(%)
Variance	95,63	96,02	95,23
Var-Diff	96,06	95,64	96,47
Wrapper	97,07	96,82	97,30

Table 6.17: Dataset 5s 128Hz 0s: comparison between the result of the three feature selection methods on CGRNN with 16 channels

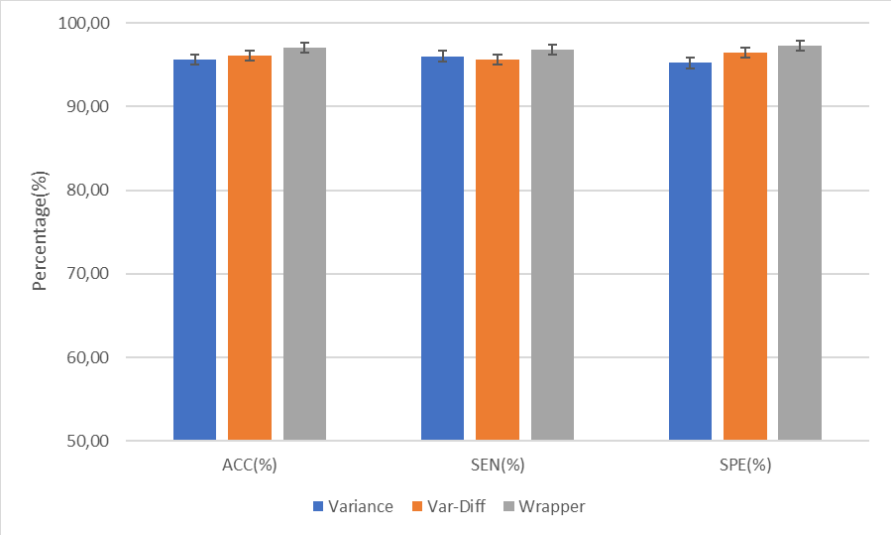


Figure 6.13: Dataset 5s 128Hz 0s: comparison between the three feature selection methods on CGRNN using the results in table 6.17

themselves, do not lead to substantial differences in the temporal features learned by the model for the purpose of predicting epileptic seizures. Even in the case of CGRNN, the learned temporal features contain useful information for prediction, but are redundant, so even if a channel is not selected by the feature selection procedure, its information is still replaced by information present in other channels, which makes the performance still good.

In any case, the feature selection methods cause a slight decrease in performance, which, although slight, shows that all the channels, in one way or another, contribute to the prediction.

Method	p-value
Wrapper-Variance	0,38
Wrapper-VarDiff	0,48
Variance-VarDiff	0,86

Table 6.18: Dataset 5s 128Hz 0s: p-value between the various feature selection methods for CGRNN

From now on, I present the results of the experiments conducted on the remaining three datasets:

- 5-second segments, sampled at 128Hz with 3 seconds of overlap;
- 5-second segments, sampled at 64Hz with no overlap;
- 5-second segments, sampled at 64Hz with 3 seconds of overlap

To avoid repetitions, the considerations will be omitted as they are very similar to what has already been reported (section 6.2). Considerations will only be added in the case of significantly different behaviors from what has already been discussed. In section 6.6, we find a final summary that takes into account and compares all the results of all the experiments from all dataset.

6.3 Dataset: 5sec segments - 128Hz - 3sec overlap

The dataset in Table 6.19 is made up of 143.940 segments, each described by 20 channels, with a length of 640 for a final shape of (143.940, 20, 640). This comes from raw EEG signals that are subsampled at 128 Hz with 3 seconds of overlap between segments

Frequency	**Overlap**	**#Segments**	**#Channels**	**Seq.Len**
128	3	143.940	20	640

Table 6.19: Dataset: 5 second segments with 3 seconds overlap and 128Hz frequency

In this experiment, unlike the previous, I have used a dataset in which the segments overlap. More precisely, the last three seconds of one segment correspond to the first three of another.
This dataset was created to verify how RNN models behave when segments share information. The idea is that overlapping segments can help RNN models better correlate segments with each other, resulting in better performance.

6.3.1 5-Fold RCV

Metrics	LSTM	CGRNN	p-value
ACC	98.90 ± 0.49	99.32 ± 0.07	0.14
SEN	98.51 ± 0.91	99.25 ± 0.33	0.18
SPE	99.28 ± 0.57	99.38 ± 0.30	0.78

Table 6.20: Dataset 5s 128Hz 3s: summary result of the RCV procedure on the test set

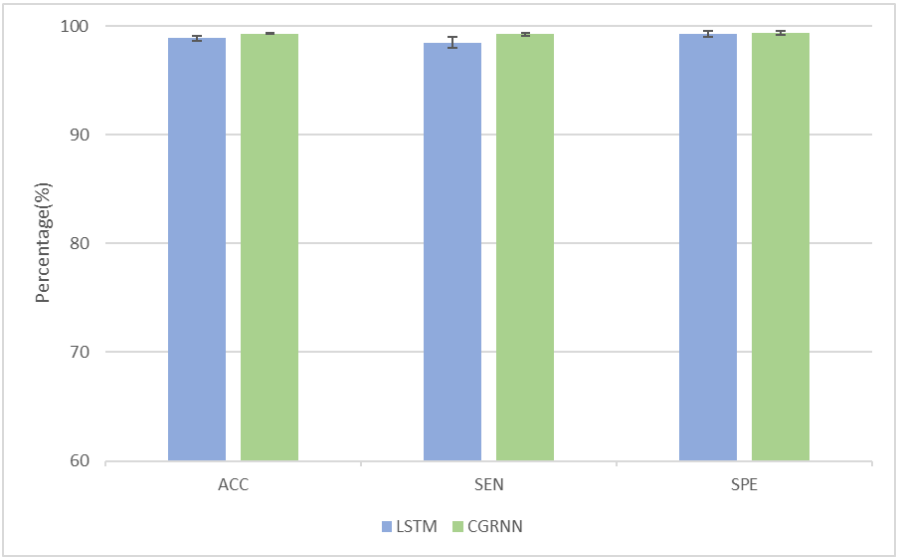


Figure 6.14: 5F RCV: comparison between LSTM and CGRNN on dataset 5s 128Hz 3 seconds overlap. The data are the one in table 6.20

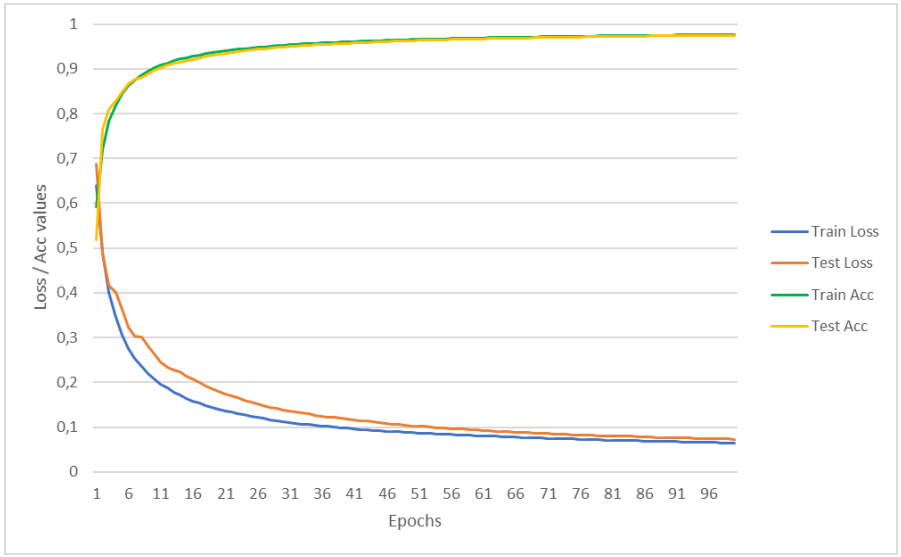


Figure 6.15: 5F RCV: training and test accuracy/loss for the LSTM model on dataset 5s 128Hz 3 seconds overlap

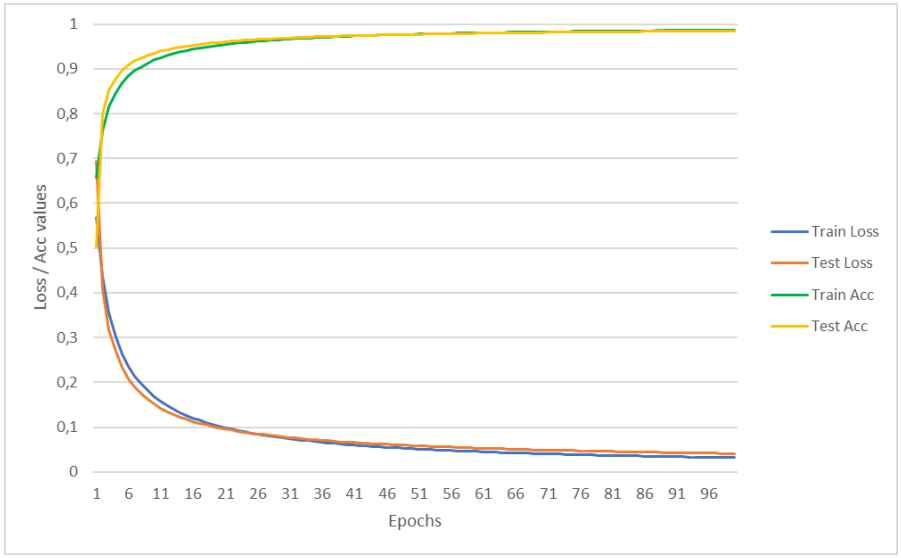


Figure 6.16: 5F RCV: training and test accuracy/loss for the CGRNN model on dataset 5s 128Hz 3 seconds overlap

6.3.2 5-Fold RCV randomized channels

Metrics	LSTM	CGRNN	p-value
ACC	98.65 ± 0.65	99.15 ± 0.47	0.26
SEN	98.47 ± 1.17	99.48 ± 0.23	0.13
SPE	99.82 ± 0.89	99.81 ± 1.1	0.98

Table 6.21: Dataset 5s 128Hz 3s: summary result of the RCV procedure with randomized channels on the test set

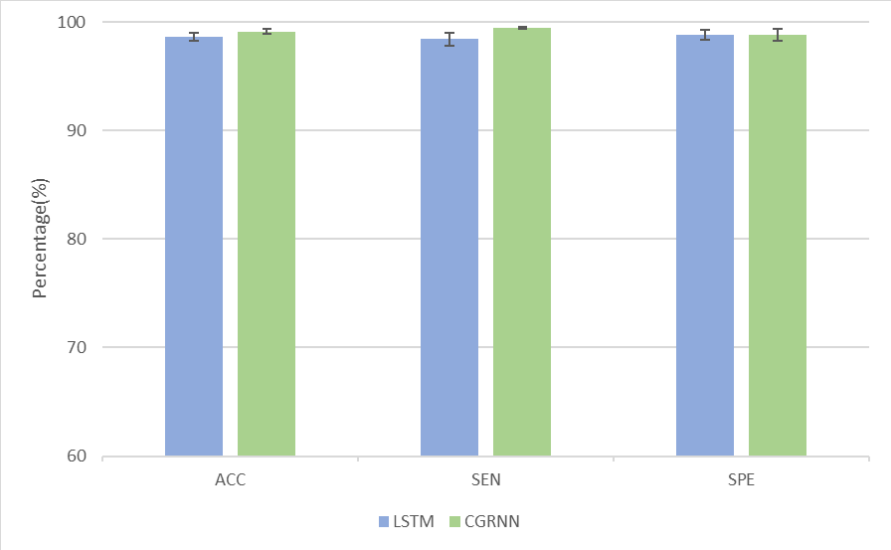


Figure 6.17: 5F RCV: comparison between LSTM and CGRNN on dataset 5s 128Hz 3 seconds overlap. The data are the one in table 6.21

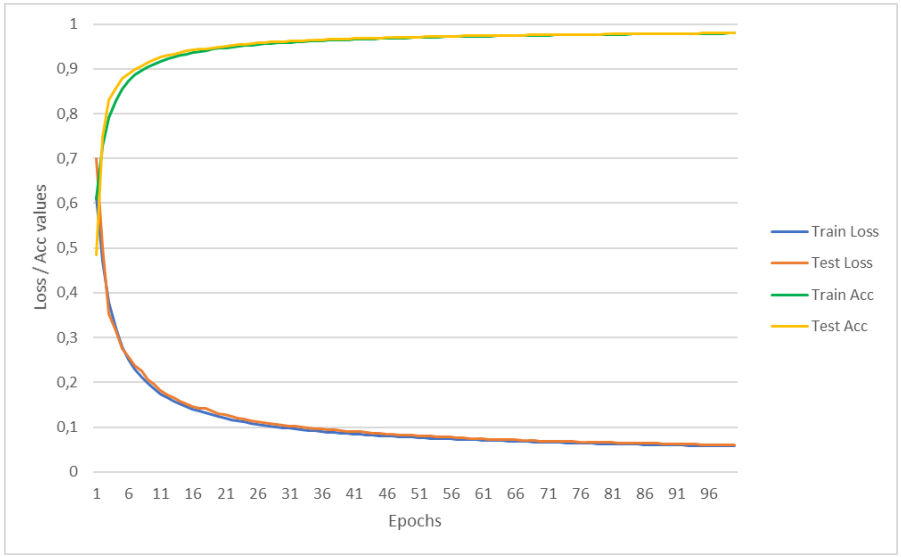


Figure 6.18: 5F RCV: training and test accuracy/loss for the LSTM model on dataset 5s 128Hz 3 seconds overlap

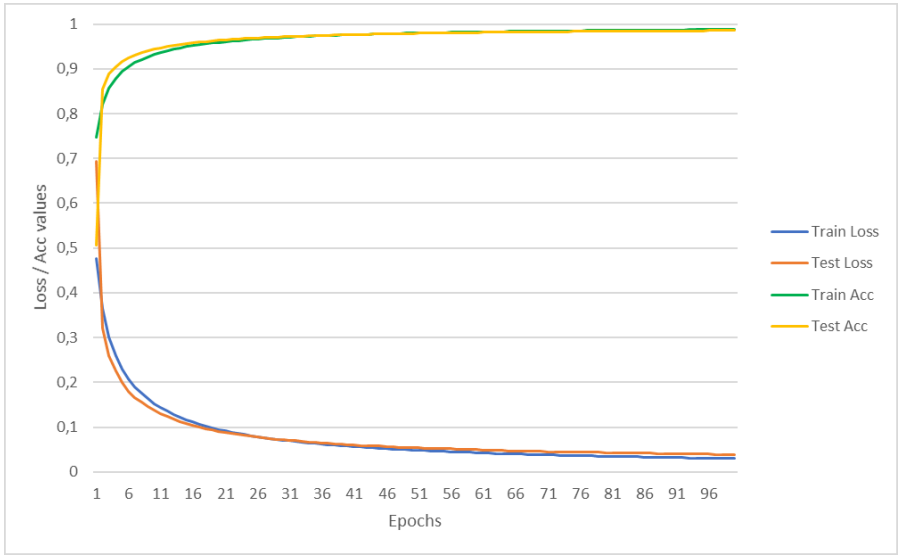


Figure 6.19: 5F RCV: training and test accuracy/loss for the CGRNN model on dataset 5s 128Hz 3 seconds overlap

6.3.3 LOO

Metrics	LSTM	CGRNN	p-value
ACC	60.25 ± 8.73	56.64 ± 7.20	0.53
SEN	63.21 ± 12.52	59.70 ± 13.37	0.71
SPE	57.28 ± 12.06	50.46 ± 11.09	0.41

Table 6.22: Dataset 5s 128Hz 3s: summary result of the LOO procedure on the test set

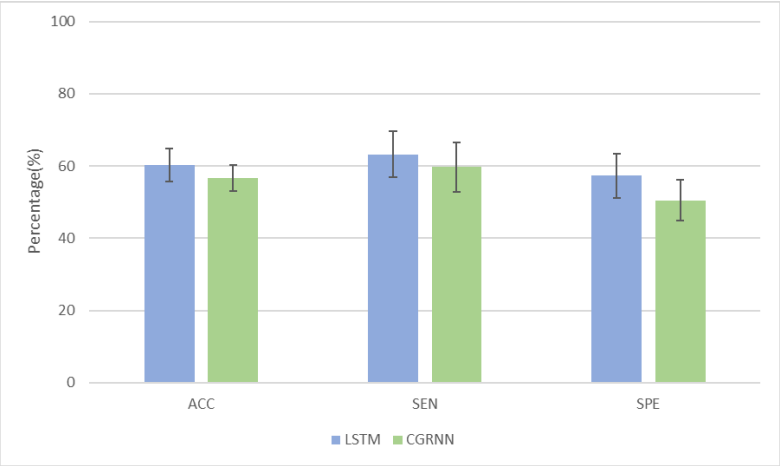


Figure 6.20: LOO: comparison between LSTM and CGRNN on dataset 5s 128Hz 3s overlap. The data are the one in table 6.22

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	1797	6703	31,19	1,56	61,00
Patient 2	898	13187	50,84	100,00	1,45
Patient 3	1798	13576	58,26	64,46	52,02
Patient 4	3598	22986	42,41	47,61	37,17
Patient 5	898	4343	73,27	58,57	88,03
Patient 6	6908	42659	95,04	96,60	93,48
Patient 7	898	4319	27,62	4,68	50,67
Patient 8	5398	27678	40,52	52,56	28,46
Patient 9	1798	6912	97,41	99,50	95,30
Patient 10	1153	7514	55,77	99,48	11,99
Patient 11	1798	8986	98,83	99,05	98,60
Patient 12	2698	35588	45,83	85,51	5,86
Patient 13	1798	5795	96,34	99,94	92,72
Patient 14	1348	4967	82,96	94,44	71,42
Patient 15	2025	9013	68,53	55,16	82,01
Patient 16	1169	9359	32,06	1,45	62,72
Patient 17	1798	8239	63,31	42,60	84,15
Patient 18	6067	47170	60,03	52,53	67,55
Patient 19	2698	14024	78,66	66,83	90,59
Patient 20	1798	16289	72,27	66,63	77,94
Patient 21	4498	41827	77,20	99,51	54,87
Patient 22	3598	17472	39,34	58,28	20,25
Patient 23	2088	15765	87,14	94,49	79,73
Patient 24	1475	9824	50,24	90,44	9,87
Patient 25	1798	7988	75,81	94,94	56,55
Patient 26	2698	21580	56,86	34,47	79,42
Patient 27	1798	14074	3,93	0,17	7,73
Patient 28	2698	19789	32,91	63,42	2,17
Patient 29	2978	9152	52,89	8,29	97,55

Table 6.23: Dataset 5s 128Hz 3s test set: results for the LOO procedure with LSTM divided by patient

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	1797	6703	19,95	22,48	17,40
Patient 2	898	13187	53,52	100,00	6,82
Patient 3	1798	13576	69,25	96,61	41,71
Patient 4	3598	22986	45,97	29,32	62,75
Patient 5	898	4343	59,99	35,08	85,01
Patient 6	6908	42659	75,09	96,42	53,39
Patient 7	898	4319	46,60	3,45	89,93
Patient 8	5398	27678	68,19	76,49	59,82
Patient 9	1798	6912	89,45	93,05	85,83
Patient 10	1153	7514	54,77	90,55	18,94
Patient 11	1798	8986	93,72	92,38	95,07
Patient 12	2698	35588	50,26	99,33	0,82
Patient 13	1798	5795	71,46	99,83	42,89
Patient 14	1348	4967	54,13	31,08	77,31
Patient 15	2025	9013	46,88	54,72	38,70
Patient 16	1169	9359	49,26	0,00	10,00
Patient 17	1798	8239	50,53	3,11	98,26
Patient 18	6067	47170	57,76	74,42	40,82
Patient 19	2698	14024	32,89	19,90	45,97
Patient 20	1798	16289	71,18	50,72	91,77
Patient 21	4498	41827	80,51	96,33	64,57
Patient 22	3598	17472	55,12	89,83	20,14
Patient 23	2088	15765	65,87	99,28	31,13
Patient 24	1475	9824	68,31	84,54	52,01
Patient 25	1798	7988	70,12	88,99	51,12
Patient 26	2698	21580	70,67	52,52	88,95
Patient 27	1798	14074	6,25	2,67	9,85
Patient 28	2698	19789	22,40	33,51	11,20
Patient 29	2978	9152	42,73	14,88	71,24

Table 6.24: Dataset 5s 128Hz 3s test set: results for the LOO procedure with CGRNN divided by patient

6.3.4 LOO randomized channels

Metrics	LSTM	CGRNN	p-value
ACC	57.89 ± 7.88	59.38 ± 8.14	0.79
SEN	60.43 ± 12.42	63.30 ± 11.96	0.74
SPE	55.34 ± 11.96	55.41 ± 11.95	0.99

Table 6.25: Dataset 5s 128Hz 3s: summary result of the LOO procedure on the test set with randomized channels

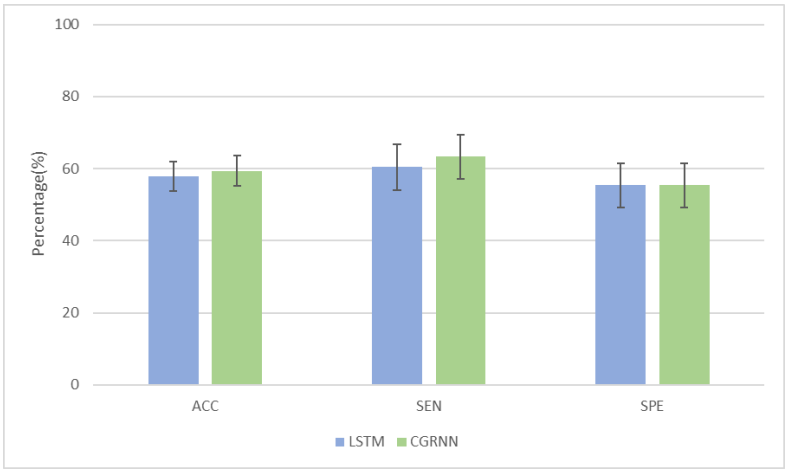


Figure 6.21: LOO with randomized channels: comparison between LSTM and CGRNN on dataset 5s 128Hz with 3 second overlap. The data are the one in table 6.25

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	1797	6703	28,63	4,17	53,22
Patient 2	898	13187	60,55	96,77	24,16
Patient 3	1798	13576	25,06	38,49	11,53
Patient 4	3598	22986	33,31	35,80	30,81
Patient 5	898	4343	69,87	61,14	78,64
Patient 6	6908	42659	87,77	83,06	92,49
Patient 7	898	4319	49,50	97,88	0,89
Patient 8	5398	27678	63,12	97,07	29,09
Patient 9	1798	6912	89,23	94,99	83,43
Patient 10	1153	7514	61,07	81,79	40,31
Patient 11	1798	8986	96,96	96,33	97,59
Patient 12	2698	35588	49,53	98,04	0,67
Patient 13	1798	5795	91,21	99,39	82,98
Patient 14	1348	4967	69,68	47,40	92,09
Patient 15	2025	9013	71,28	44,94	97,86
Patient 16	1169	9359	43,19	7,10	79,35
Patient 17	1798	8239	65,90	44,27	87,68
Patient 18	6067	47170	56,65	45,33	67,98
Patient 19	2698	14024	49,94	0,59	99,66
Patient 20	1798	16289	68,50	50,56	86,56
Patient 21	4498	41827	65,55	99,78	31,29
Patient 22	3598	17472	48,54	67,09	29,83
Patient 23	2088	15765	79,11	82,61	75,58
Patient 24	1475	9824	59,04	82,78	35,19
Patient 25	1798	7988	62,78	92,66	32,70
Patient 26	2698	21580	62,02	60,82	63,22
Patient 27	1798	14074	2,23	1,39	3,08
Patient 28	2698	19789	26,10	29,65	22,52
Patient 29	2978	9152	42,52	10,61	74,48

Table 6.26: Dataset 5s 128Hz 3s overlap with randomized channels on test set: results for the LOO procedure with LSTM divided by patient

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	1797	6703	26,09	46,69	5,37
Patient 2	898	13187	54,52	99,78	9,06
Patient 3	1798	13576	77,62	92,94	62,21
Patient 4	3598	22986	41,17	14,62	67,93
Patient 5	898	4343	69,03	50,45	87,70
Patient 6	6908	42659	91,22	96,64	85,71
Patient 7	898	4319	42,75	64,37	21,03
Patient 8	5398	27678	60,64	54,24	67,09
Patient 9	1798	6912	85,07	79,53	90,65
Patient 10	1153	7514	73,39	90,11	56,65
Patient 11	1798	8986	96,93	96,38	97,48
Patient 12	2698	35588	50,00	99,48	0,15
Patient 13	1798	5795	81,08	100,00	62,04
Patient 14	1348	4967	58,78	45,47	72,16
Patient 15	2025	9013	63,41	92,05	33,56
Patient 16	1169	9359	24,61	6,59	43,17
Patient 17	1798	8239	84,15	76,42	91,94
Patient 18	6067	47170	63,45	74,44	52,27
Patient 19	2698	14024	46,54	30,36	62,85
Patient 20	1798	16289	73,33	50,17	96,64
Patient 21	4498	41827	84,90	98,22	71,47
Patient 22	3598	17472	37,96	62,42	13,31
Patient 23	2088	15765	64,79	99,33	28,88
Patient 24	1475	9824	67,87	91,05	44,59
Patient 25	1798	7988	72,27	59,79	84,83
Patient 26	2698	21580	49,81	0,00	100,00
Patient 27	1798	14074	4,99	2,95	7,05
Patient 28	2698	19789	22,99	43,51	2,32
Patient 29	2978	9152	52,89	17,76	88,83

Table 6.27: Dataset 5s 128Hz 3s overlap with randomized channels on test set: results for the LOO procedure with CGRNN divided by patient

6.3.5 Filter feature selection

Variance		ACC(%)	SEN(%)	SPE(%)
1 channel	**LSTM**	74,64	67,94	81,33
	CGRNN	82,29	79,10	97,40
3 channels	**LSTM**	89,65	87,03	92,22
	CGRNN	90,16	82,74	94,17
5 channels	**LSTM**	92,02	88,70	95,26
	CGRNN	96,86	88,95	99,06
10 channels	**LSTM**	96,58	98,09	95,08
	CGRNN	95,02	88,75	96,49
15 channels	**LSTM**	98,02	98,62	97,39
	CGRNN	99,28	95,20	99,18
16 channels	**LSTM**	—	—	—
	CGRNN	98,94	98,13	99,40

Table 6.28: Dataset 5s 128Hz 3s overlap: results for the feature selection procedure using the variance filter method.

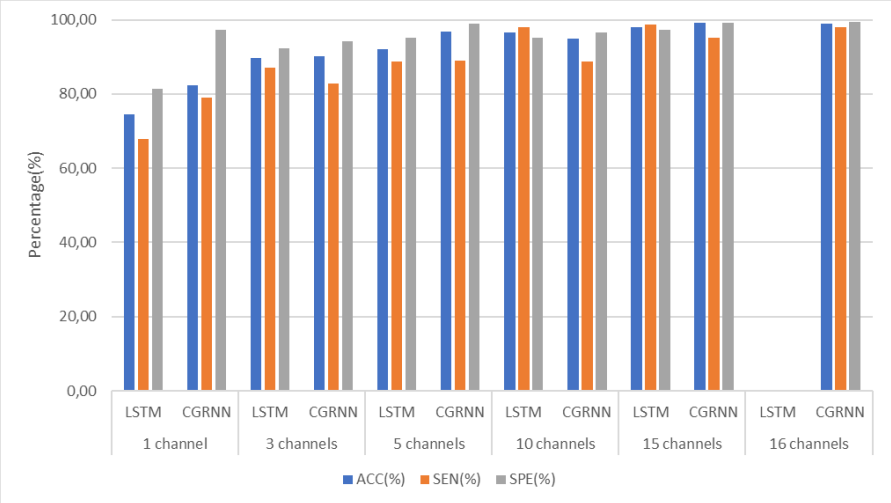


Figure 6.22: Dataset 5s 128Hz 3s overlap: result for the variance feature selection based on the results on table 6.28

Variance difference		ACC(%)	SEN(%)	SPE(%)
1 channel	**LSTM**	70,64	61,24	78,33
	CGRNN	82,91	85,44	92,60
3 channels	**LSTM**	89,62	93,00	86,22
	CGRNN	91,87	83,08	96,22
5 channels	**LSTM**	93,25	94,09	92,41
	CGRNN	97,06	91,35	99,51
10 channels	**LSTM**	97,97	96,89	99,04
	CGRNN	95,21	94,97	99,38
15 channels	**LSTM**	95,68	92,34	99,15
	CGRNN	97,94	98,55	98,71
16 channels	**LSTM**	—	—	—
	CGRNN	98,33	95,94	97,27

Table 6.29: Dataset 5s 128Hz 3s overlap: results for the feature selection procedure using the variance difference filter method.

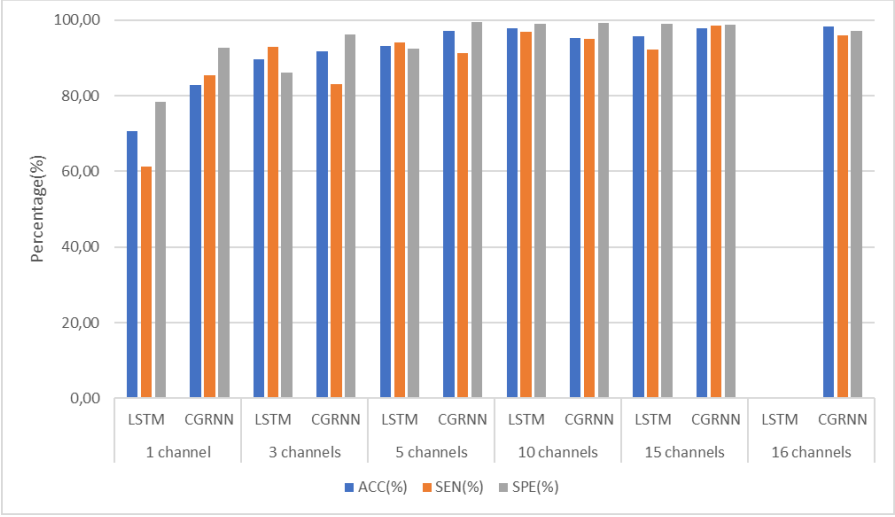


Figure 6.23: Dataset 5s 128Hz 3s overlap: result for the variance difference feature selection based on the results on table 6.29

6.3.6 Wrapper feature selection

Metrics	LSTM	CGRNN	p-value
ACC	97.92 ± 0.88	97.87 ± 0.38	0.96
SEN	98.51 ± 0.97	97.15 ± 0.65	0.11
SPE	97.48 ± 1.05	98.59 ± 0.63	0.27

Table 6.30: Dataset 5s 128Hz 3s: summary result of the wrapper feature selection method presented in listing 4.1

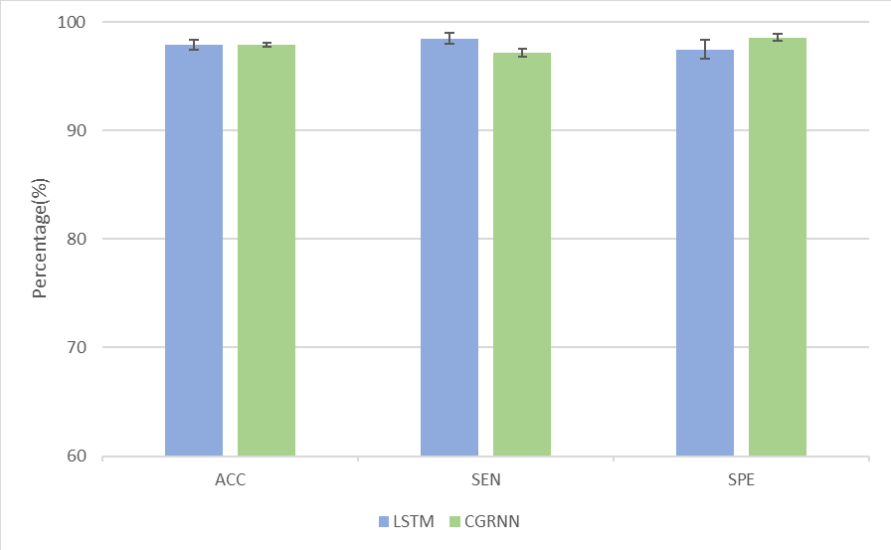


Figure 6.24: Dataset 5s 128Hz 3s: result for the wrapper feature selection method on the results on table 6.30

Method	ACC(%)	SEN(%)	SPE(%)
Variance	98,02	97,37	98,62
Var-Diff	98,08	98,81	97,38
Wrapper	97,92	98,51	97,48

Table 6.31: Dataset 5s 128Hz 3s: comparison between the result of the three feature selection methods on LSTM with 15 channels

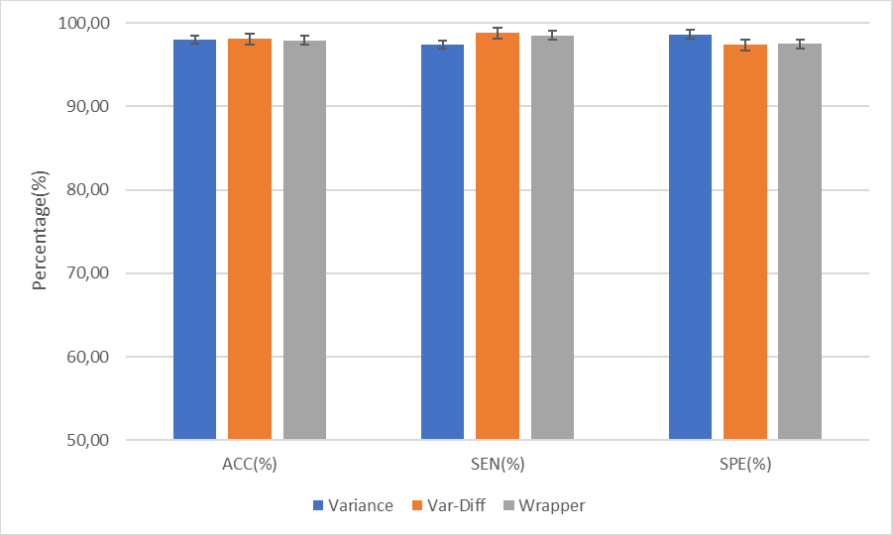


Figure 6.25: Dataset 5s 128Hz 3s: comparison between the three feature selection methods on LSTM using the results in table 6.31

Method	p-value
Wrapper-Variance	0,99
Wrapper-VarDiff	0,65
Variance-VarDiff	0,68

Table 6.32: Dataset 5s 128Hz 0s: p-value between the various feature selection methods for LSTM on accuracy metric

Method	ACC(%)	SEN(%)	SPE(%)
Variance	98,92	98,71	99,18
Var-Diff	98,33	97,27	99,40
Wrapper	97,90	97,19	98,60

Table 6.33: Dataset 5s 128Hz 3s: comparison between the result of the three feature selection methods on CGRNN with 16 channels

Method	p-value
Wrapper-Variance	0,81
Wrapper-VarDiff	76
Variance-VarDiff	0,95

Table 6.34: Dataset 5s 128Hz 3s: p-value between the various feature selection methods for CGRNN on accuracy metric

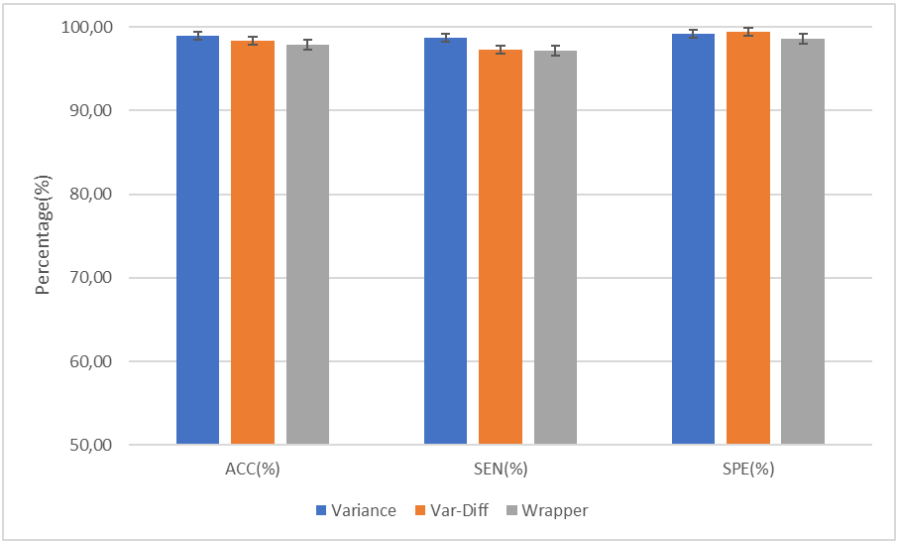


Figure 6.26: Dataset 5s 128Hz 3s: comparison between the three feature selection methods on CGRNN using the results in table 6.33

6.4 Dataset: 5sec segments - 64Hz - no overlap

The dataset in table 6.35 is composed of 57.612 segments, each described by 20 channels, with a length of 320 for a final shape of (57.612, 20, 320). This comes from the raw EEG signals that are subsampled at 64 Hz without overlap between segments.

Frequency	**Overlap**	**#Segments**	**#Channels**	**Seq.Len**
64	0	57.612	20	320

Table 6.35: Dataset: 5 second segments with no overlap and 64Hz frequency

In this experiment, the sampling rate was halved from 128Hz to 64Hz, reducing the length of the segments from 640 to 320. This change was made because RNNs work better on smaller sequences.
The goal of the experiment is to see if a length of 640 is enough to learn temporal features, or if a reduced length could lead to better performance. In the latter case, it could be concluded that a length of 640 is excessive for the models used.
It should be noted that reducing the sampling rate results in a reduction in the length of the segments, as the amount of information contained in each segment is halved.

6.4.1 5-Fold RCV

Metrics	**LSTM**	**CGRNN**	**p-value**
ACC	98.02 ± 0.83	97.26 ± 0.83	0.24
SEN	97.73 ± 1.98	97.05 ± 0.43	0.53
SPE	98.31 ± 0.67	97.62 ± 1.35	0.38

Table 6.36: Dataset 5s 64Hz no overlap: summary result of the RCV procedure on the test set

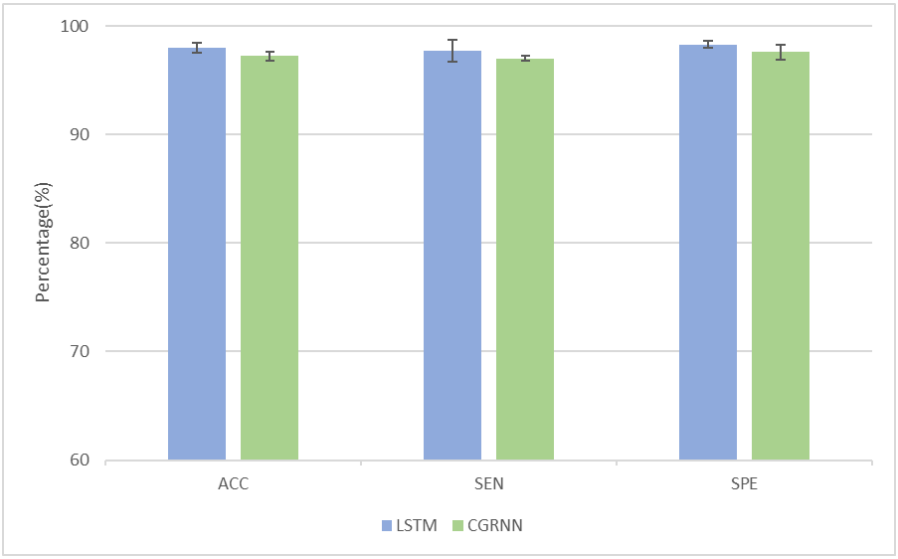


Figure 6.27: 5F RCV: comparison between LSTM and CGRNN on dataset 5s 64Hz with no overlap. The data are the one in table 6.36

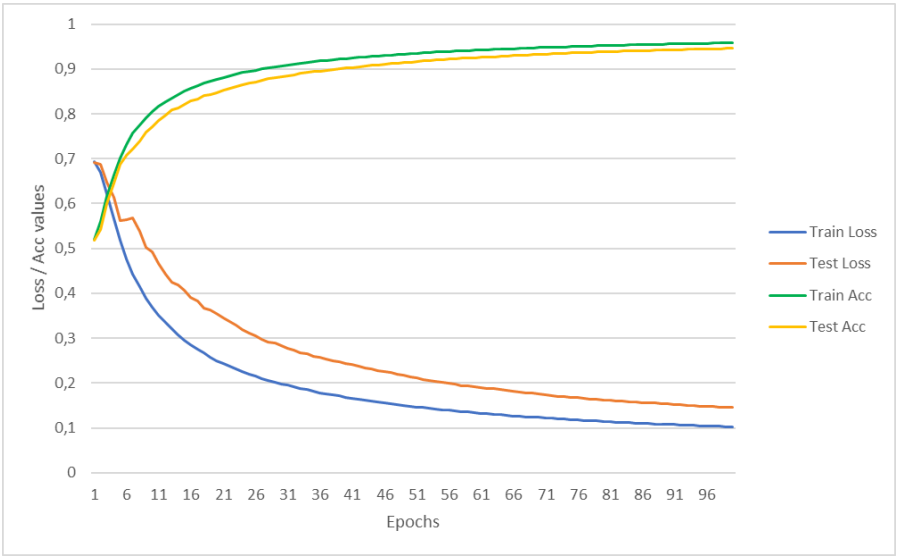


Figure 6.28: 5F RCV: training and test accuracy/loss for the LSTM model on dataset 5s 64Hz no overlap

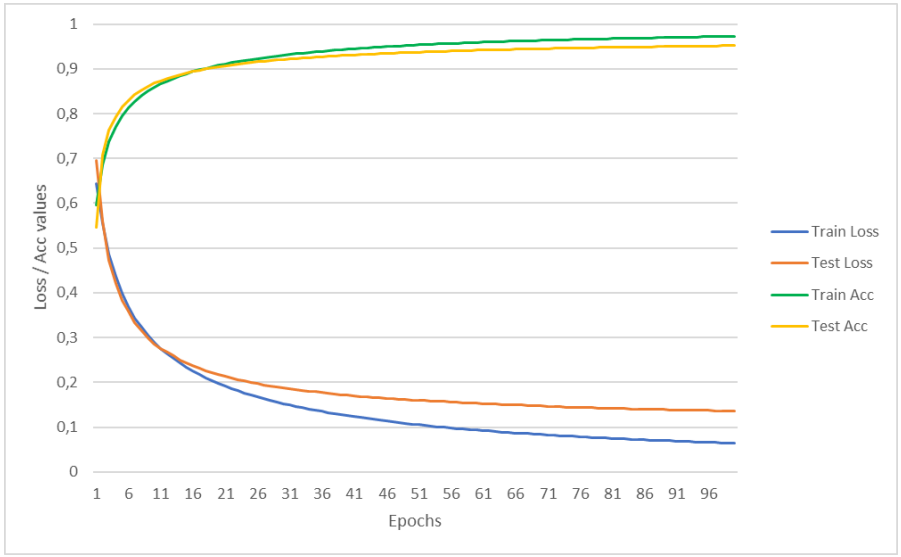


Figure 6.29: 5F RCV: training and test accuracy/loss for the CGRNN model on dataset 5s 64Hz no overlap

6.4.2 5-Fold RCV randomized channels

Metrics	LSTM	CGRNN	p-value
ACC	97.78 ± 0.82	96.61 ± 0.95	0.11
SEN	96.69 ± 2.01	96.12 ± 1.73	0.72
SPE	98.85 ± 0.84	97.26 ± 0.91	0.36

Table 6.37: Dataset 5s 64Hz no overlap: summary result of the RCV procedure on the test set with randomized channels

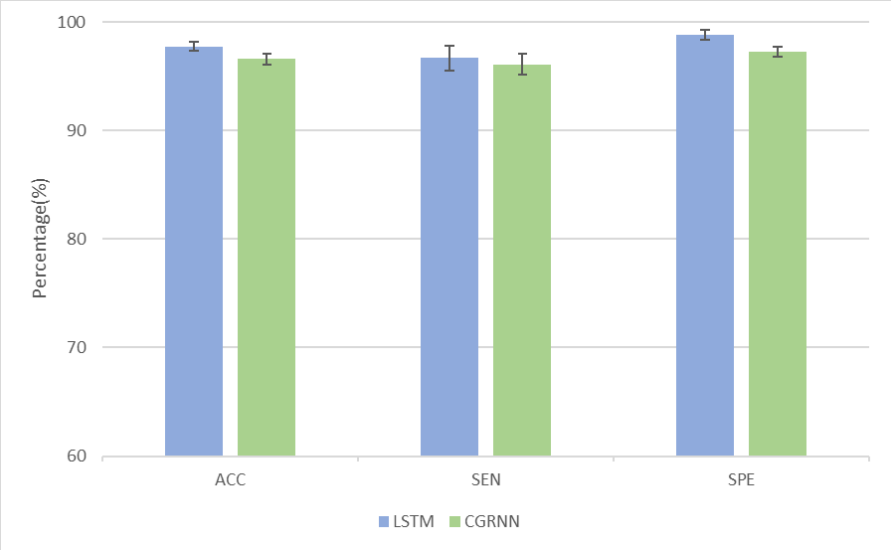


Figure 6.30: 5F RCV: comparison between LSTM and CGRNN on dataset 5s 64Hz with no overlap and randomized channels. The data are the one in table 6.37

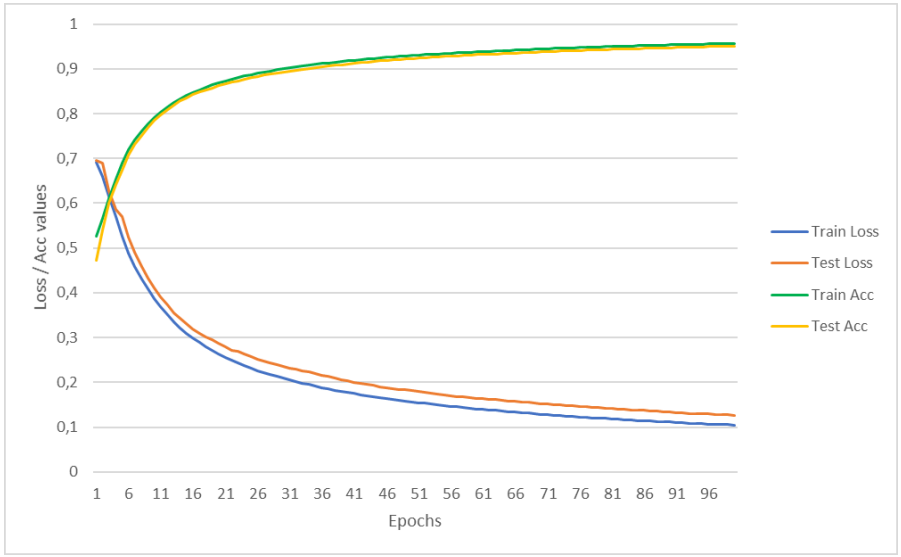


Figure 6.31: 5F RCV: training and test accuracy/loss for the LSTM model on dataset 5s 64Hz no overlap with randomized channels

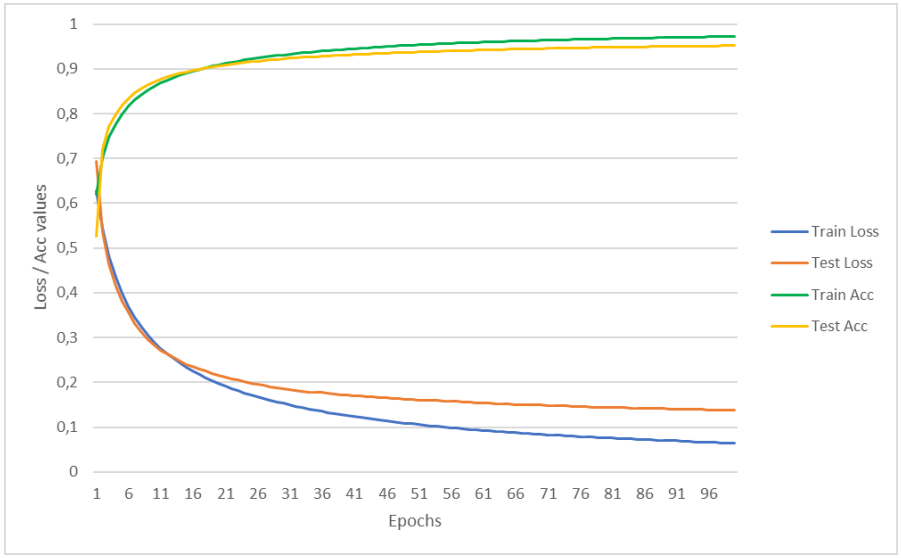


Figure 6.32: 5F RCV: training and test accuracy/loss for the CGRNN model on dataset 5s 64Hz no overlap with randomized channels

6.4.3 LOO

Metrics	LSTM	CGRNN	p-value
ACC	60.25 ± 8.58	52.74 ± 6.11	0.17
SEN	65.19 ± 12.12	34.05 ± 13.10	0.02
SPE	55.18 ± 10.41	73.79 ± 9.90	0.01

Table 6.38: Dataset 5s 64Hz 0s: summary result of the LOO procedure on the test set

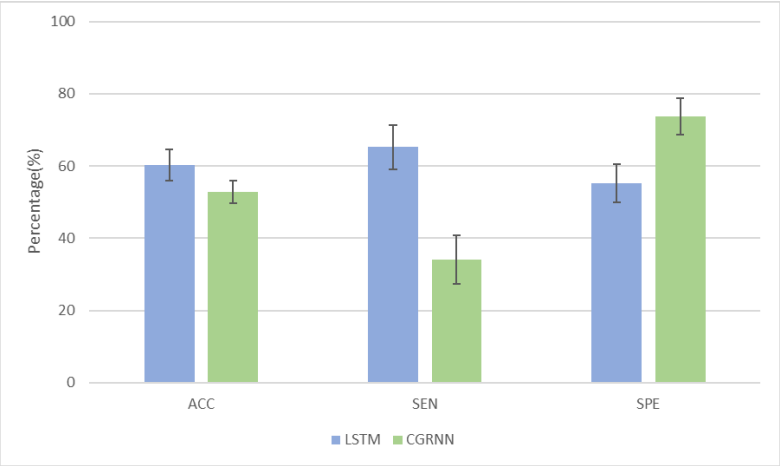


Figure 6.33: LOO: comparison between LSTM and CGRNN on dataset 5s 64Hz no overlap. The data are the one in table 6.38

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	719	2681	22,73	5,98	40,20
Patient 2	360	5275	53,98	85,56	20,93
Patient 3	720	5431	77,57	84,44	70,69
Patient 4	1440	9195	25,31	11,04	39,58
Patient 5	360	1738	70,31	86,39	53,49
Patient 6	2763	17064	77,80	93,85	61,62
Patient 7	360	1728	50,85	90,83	9,01
Patient 8	2160	11072	56,44	79,40	33,47
Patient 9	720	2765	88,33	97,36	79,31
Patient 10	462	3006	66,07	75,97	55,53
Patient 11	720	3595	92,64	94,72	90,56
Patient 12	1080	14235	50,23	99,63	0,09
Patient 13	720	2318	94,93	98,89	90,97
Patient 14	540	1987	64,39	47,59	81,98
Patient 15	810	3606	75,69	56,67	95,19
Patient 16	468	3744	32,00	9,83	54,57
Patient 17	720	3296	83,06	84,58	81,53
Patient 18	2427	18868	62,79	72,31	53,18
Patient 19	1080	5610	75,47	55,65	95,58
Patient 20	720	6516	53,06	52,50	53,61
Patient 21	1800	16731	91,27	98,67	83,80
Patient 22	1440	6989	60,31	92,78	27,85
Patient 23	835	6306	81,73	99,16	64,17
Patient 24	590	3930	55,99	88,14	22,24
Patient 25	720	3196	65,28	56,81	73,75
Patient 26	1080	8632	59,24	40,00	78,76
Patient 27	720	5630	2,43	0,69	4,17
Patient 28	1080	7916	17,54	15,46	19,64
Patient 29	1192	3661	39,91	15,77	64,37

Table 6.39: Dataset 5s 64Hz no overlap on test set: results for the LOO procedure with LSTM divided by patient

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	719	2681	16,90	14,88	19,01
Patient 2	360	5275	43,75	0,00	100,00
Patient 3	720	5431	79,33	91,25	66,86
Patient 4	1440	9195	48,86	0,00	100,00
Patient 5	360	1738	43,75	0,00	100,00
Patient 6	2763	17064	49,80	0,00	100,00
Patient 7	360	1728	43,75	0,00	100,00
Patient 8	2160	11072	55,92	77,22	33,62
Patient 9	720	2765	83,31	71,25	95,93
Patient 10	462	3006	60,38	70,35	49,77
Patient 11	720	3595	48,86	0,00	100,00
Patient 12	1080	14235	47,27	0,00	100,00
Patient 13	720	2318	48,86	0,00	100,00
Patient 14	540	1987	47,27	0,00	100,00
Patient 15	810	3606	47,27	0,00	100,00
Patient 16	468	3744	28,57	10,26	48,60
Patient 17	720	3296	61,72	72,78	50,15
Patient 18	2427	18868	67,10	58,80	75,83
Patient 19	1080	5610	42,24	30,65	55,17
Patient 20	720	6516	71,80	50,69	93,90
Patient 21	1800	16731	49,78	0,00	100,00
Patient 22	1440	6989	80,54	91,67	68,90
Patient 23	835	6306	70,97	98,32	43,43
Patient 24	590	3930	64,93	68,14	61,57
Patient 25	720	3196	59,45	78,61	39,39
Patient 26	1080	8632	73,68	52,78	97,00
Patient 27	720	5630	31,53	12,08	51,89
Patient 28	1080	7916	26,07	3,33	51,45
Patient 29	1192	3661	36,11	34,65	37,68

Table 6.40: Dataset 5s 64Hz no overlap on test set: results for the LOO procedure with CGRNN divided by patient

6.4.4 LOO randomized channels

Metrics	LSTM	CGRNN	p-value
ACC	58.35 ± 8.45	52.69 ± 7.21	0.32
SEN	61.90 ± 12.48	41.85 ± 13.64	0.04
SPE	54.64 ± 12.98	64.16 ± 12.11	0.29

Table 6.41: Dataset 5s 64Hz 0s: summary result of the LOO procedure on the test set with randomized channels

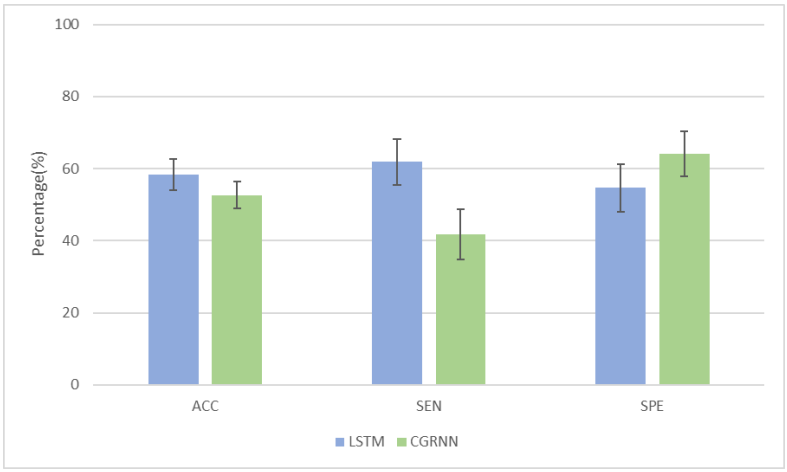


Figure 6.34: LOO: comparison between LSTM and CGRNN on dataset 5s 64Hz no overlap with randomized channels. The data are the one in table 6.41

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	719	2681	22,23	40,06	3,63
Patient 2	360	5275	54,69	99,72	7,56
Patient 3	720	5431	62,50	75,69	49,31
Patient 4	1440	9195	47,88	3,68	92,08
Patient 5	360	1738	68,18	84,17	51,45
Patient 6	2763	17064	93,62	95,01	92,23
Patient 7	360	1728	26,14	42,22	9,30
Patient 8	2160	11072	51,20	83,47	18,94
Patient 9	720	2765	90,07	94,72	85,42
Patient 10	462	3006	58,04	93,51	20,28
Patient 11	720	3595	97,22	96,11	98,33
Patient 12	1080	14235	50,23	99,63	0,09
Patient 13	720	2318	94,72	91,25	98,19
Patient 14	540	1987	50,19	4,26	98,26
Patient 15	810	3606	70,75	45,43	96,71
Patient 16	468	3744	28,88	1,07	57,17
Patient 17	720	3296	69,03	52,92	85,14
Patient 18	2427	18868	74,07	58,18	90,10
Patient 19	1080	5610	76,07	70,28	81,95
Patient 20	720	6516	73,54	50,00	97,08
Patient 21	1800	16731	71,65	99,11	43,95
Patient 22	1440	6989	51,28	75,14	27,43
Patient 23	835	6306	69,53	98,92	39,93
Patient 24	590	3930	60,94	67,12	54,45
Patient 25	720	3196	48,26	86,39	10,14
Patient 26	1080	8632	62,59	58,61	66,64
Patient 27	720	5630	4,44	2,22	6,67
Patient 28	1080	7916	15,49	10,37	20,68
Patient 29	1192	3661	48,48	15,86	81,55

Table 6.42: Dataset 5s 64Hz no overlap on test set with randomized channels: results for the LOO procedure with LSTM divided by patient

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	719	2681	17,83	30,04	5,08
Patient 2	360	5275	67,97	95,00	33,21
Patient 3	720	5431	48,86	0,00	100,00
Patient 4	1440	9195	29,90	26,18	33,79
Patient 5	360	1738	45,16	8,06	92,86
Patient 6	2763	17064	85,57	80,42	90,77
Patient 7	360	1728	44,22	33,33	58,21
Patient 8	2160	11072	48,86	0,00	100,00
Patient 9	720	2765	48,86	0,00	100,00
Patient 10	462	3006	50,56	64,94	35,25
Patient 11	720	3595	93,54	96,39	90,55
Patient 12	1080	14235	56,49	99,63	8,37
Patient 13	720	2318	71,45	98,06	43,60
Patient 14	540	1987	10,74	15,56	5,37
Patient 15	810	3606	84,90	85,31	84,44
Patient 16	468	3744	27,12	15,60	39,72
Patient 17	720	3296	74,79	65,28	84,74
Patient 18	2427	18868	48,75	0,00	100,00
Patient 19	1080	5610	34,77	23,61	47,21
Patient 20	720	6516	78,55	70,42	87,06
Patient 21	1800	16731	49,78	0,00	100,00
Patient 22	1440	6989	67,58	82,36	52,11
Patient 23	835	6306	49,82	0,00	100,00
Patient 24	590	3930	45,49	73,22	16,37
Patient 25	720	3196	54,83	81,53	26,89
Patient 26	1080	8632	62,79	47,31	80,06
Patient 27	720	5630	33,88	4,72	64,39
Patient 28	1080	7916	47,27	0,00	100,00
Patient 29	1192	3661	47,70	16,95	80,67

Table 6.43: Dataset 5s 64Hz no overlap on test set with randomized channels: results for the LOO procedure with CGRNN divided by patient

6.4.5 Filter feature selection

Variance		ACC(%)	SEN(%)	SPE(%)
1 channel	**LSTM**	74,64	81,33	67,94
	CGRNN	49,97	0,00	100,00
3 channels	**LSTM**	89,65	92,22	87,03
	CGRNN	75,38	80,10	70,81
5 channels	**LSTM**	92,02	95,26	88,70
	CGRNN	80,90	77,11	84,73
10 channels	**LSTM**	96,58	95,08	98,09
	CGRNN	87,59	90,39	84,79
15 channels	**LSTM**	98,02	97,39	98,62
	CGRNN	93,96	95,03	92,85
16 channels	**LSTM**	—	—	—
	CGRNN	94,87	96,02	93,67

Table 6.44: Dataset 5s 64Hz no overlap: results for the feature selection procedure using the variance filter method.

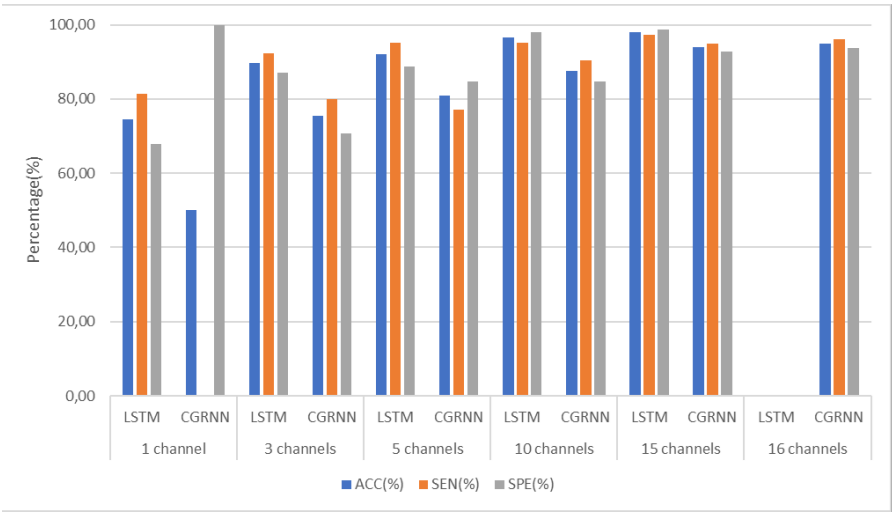


Figure 6.35: Dataset 5s 64Hz no overlap: result for the variance feature selection based on the results on table 6.44

Variance Difference		ACC(%)	SEN(%)	SPE(%)
1 channel	**LSTM**	49,35	0,00	100,00
	CGRNN	67,40	69,79	65,00
3 channels	**LSTM**	89,62	86,22	93,00
	CGRNN	74,32	74,87	73,77
5 channels	**LSTM**	93,25	92,41	94,09
	CGRNN	84,29	80,95	87,71
10 channels	**LSTM**	97,97	99,04	96,89
	CGRNN	89,88	92,39	87,44
15 channels	**LSTM**	98,09	98,81	97,38
	CGRNN	93,70	93,36	94,03
16 channels	**LSTM**	—	—	—
	CGRNN	95,26	93,78	96,77

Table 6.45: Dataset 5s 64Hz no overlap: results for the feature selection procedure using the variance difference filter method.

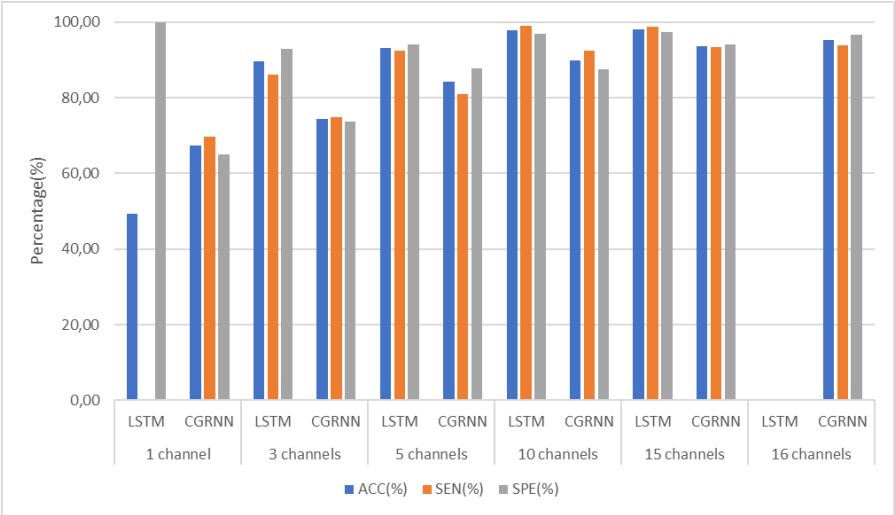


Figure 6.36: Dataset 5s 64Hz no overlap: result for the variance difference feature selection based on the results on table 6.45

6.4.6 Wrapper feature selection

Metrics	LSTM	CGRNN	p-value
ACC	97.42 ± 0.35	95.69 ± 0.38	0.16
SEN	97.16 ± 1.17	95.19 ± 1.45	0.09
SPE	97.68 ± 1.32	96.49 ± 1.21	0.23

Table 6.46: Dataset 5s 64Hz no overlap: summary result of the wrapper feature selection method presented in listing 4.1

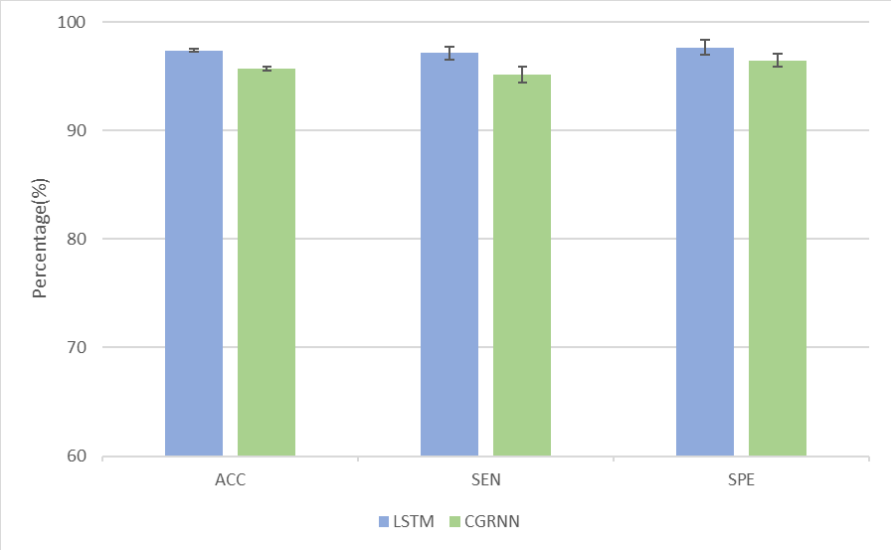


Figure 6.37: Dataset 5s 64Hz no overlap: result for the wrapper feature selection method on the results on table 6.46

Method	ACC(%)	SEN(%)	SPE(%)
Variance	97,43	97,76	97,09
Var-Diff	95,78	93,71	97,77
Wrapper	97,42	97,16	97,68

Table 6.47: Dataset 5s 64Hz no overlap: comparison between the result of the three feature selection methods on LSTM with 15 channels

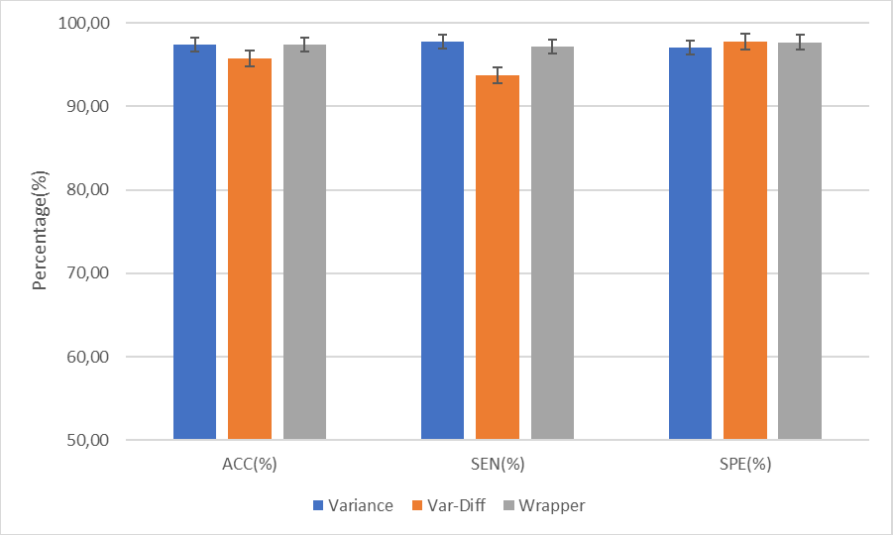


Figure 6.38: Dataset 5s 64Hz 0s: comparison between the three feature selection methods on LSTM using the results in table 6.47

Method	p-value
Wrapper-Variance	0,57
Wrapper-VarDiff	0,68
Variance-VarDiff	0,89

Table 6.48: Dataset 5s 64Hz 0s: p-value between the various feature selection methods for LSTM on accuracy metric

Method	ACC(%)	SEN(%)	SPE(%)
Variance	94,88	93,67	96,03
Var-Diff	95,26	96,77	93,78
Wrapper	95,70	95,20	96,49

Table 6.49: Dataset 5s 64Hz no overlap: comparison between the result of the three feature selection methods on CGRNN with 15 channels

Method	p-value
Wrapper-Variance	0,67
Wrapper-VarDiff	0,90
Variance-VarDiff	0,60

Table 6.50: Dataset 5s 64Hz 0s: p-value between the various feature selection methods for LSTM on accuracy metric

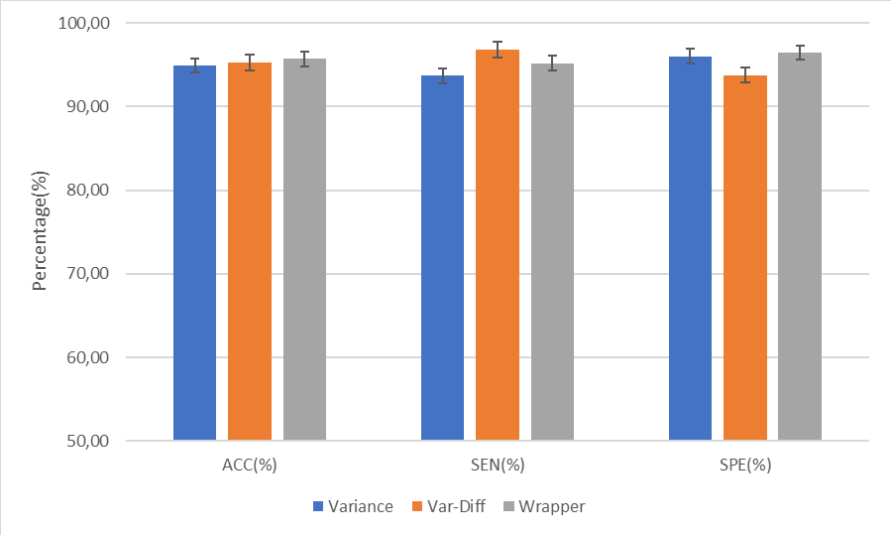


Figure 6.39: Dataset 5s 64Hz 0s: comparison between the three feature selection methods on CGRNN using the results in table 6.49

6.5 Dataset: 5sec segments - 64Hz - 3sec overlap

The dataset in table 6.51 is composed of 143.940 segments, each described by 20 channels, with a length of 320 for a final shape of (143.940, 20, 320). This comes from the raw EEG signals that are subsampled at 64 Hz with 3 seconds of overlap between segments.

Frequency	Overlap	#Segments	#Channels	Seq.Len
64	3	143.940	20	32

Table 6.51: Dataset: 5 second segments with 3 second overlap and 64Hz frequency

6.5.1 5-Fold RCV

Metrics	LSTM	CGRNN	p-value
ACC	98.89 ± 0.35	98.90 ± 0.21	0.95
SEN	98.84 ± 0.66	98.91 ± 0.61	0.90
SPE	98.94 ± 1.10	98.88 ± 0.59	0.92

Table 6.52: Dataset 5s 64Hz 3s overlap : summary result of the RCV procedure on the test set

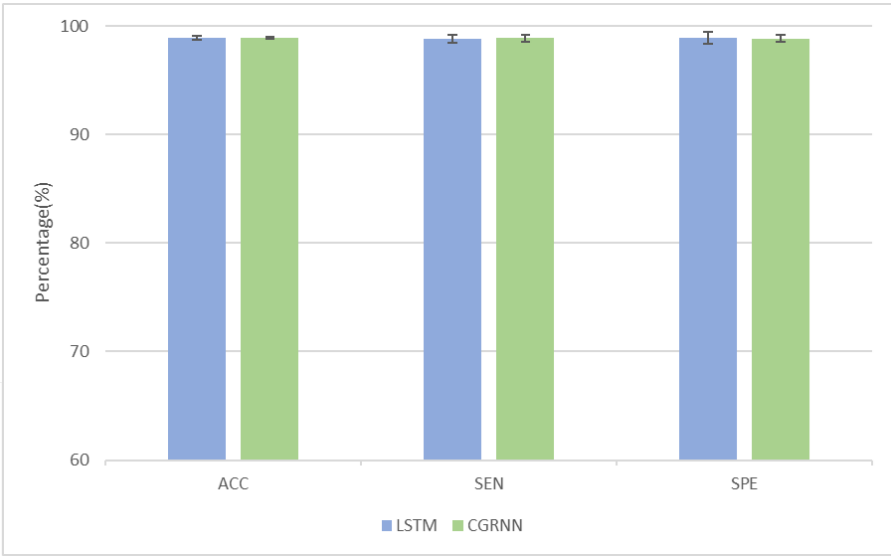


Figure 6.40: 5F RCV: comparison between LSTM and CGRNN on dataset 5s 64Hz with 3 second overlap. The data are the one in table 6.52

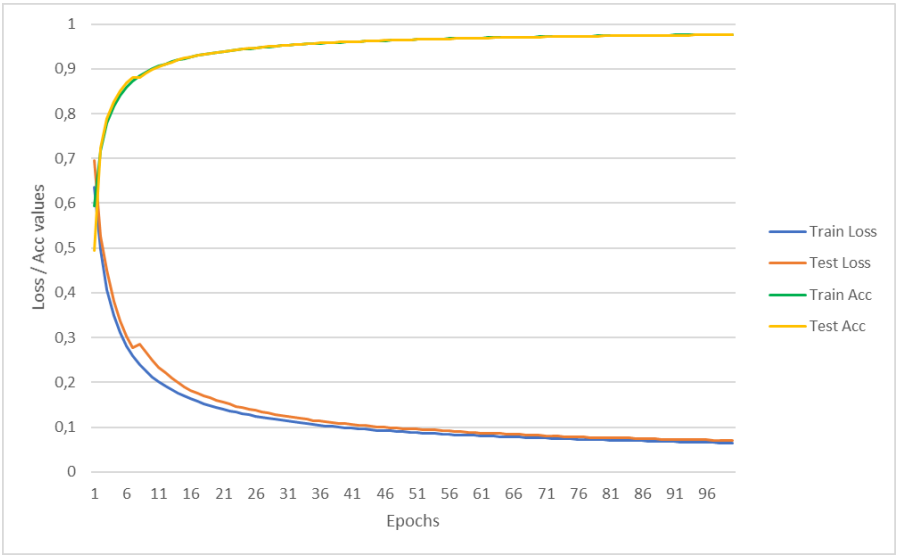


Figure 6.41: 5F RCV: training and test accuracy/loss for the LSTM model on dataset 5s 64Hz 3s overlap

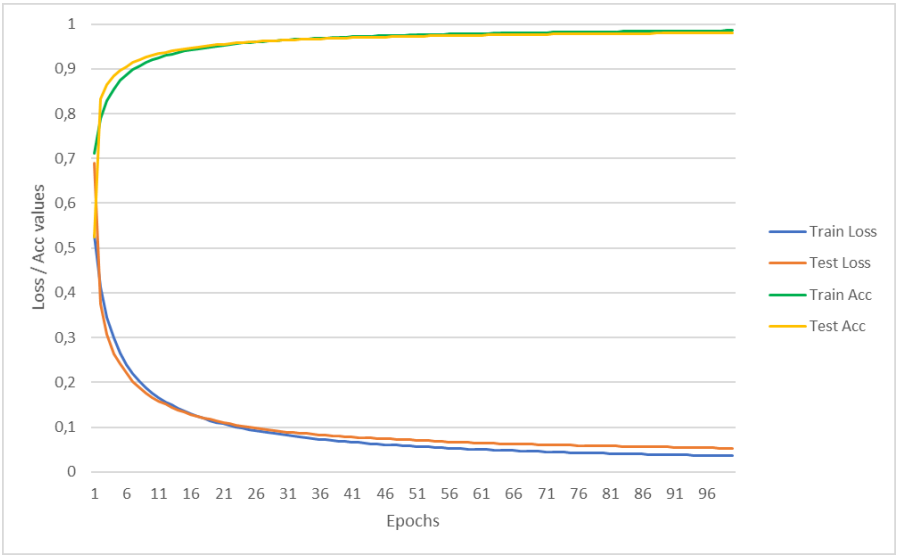


Figure 6.42: 5F RCV: training and test accuracy/loss for the CGRNN model on dataset 5s 64Hz 3s overlap

6.5.2 5-Fold RCV randomized channels

Metrics	LSTM	CGRNN	p-value
ACC	98.92 ± 0.60	99.06 ± 0.18	0.68
SEN	98.37 ± 1.37	99.57 ± 0.12	0.12
SPE	99.48 ± 0.36	98.54 ± 0.39	0.57

Table 6.53: Dataset 5s 64Hz 3s overlap with randomized channels: summary result of the RCV procedure on the test set

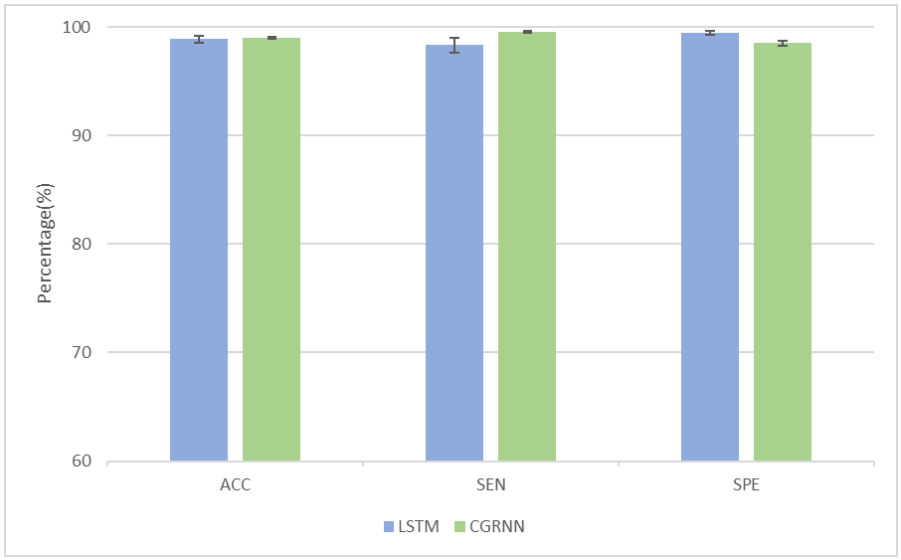


Figure 6.43: 5F RCV: comparison between LSTM and CGRNN on dataset 5s 64Hz with 3 second overlap and randomized channels. The data are the one in table 6.52

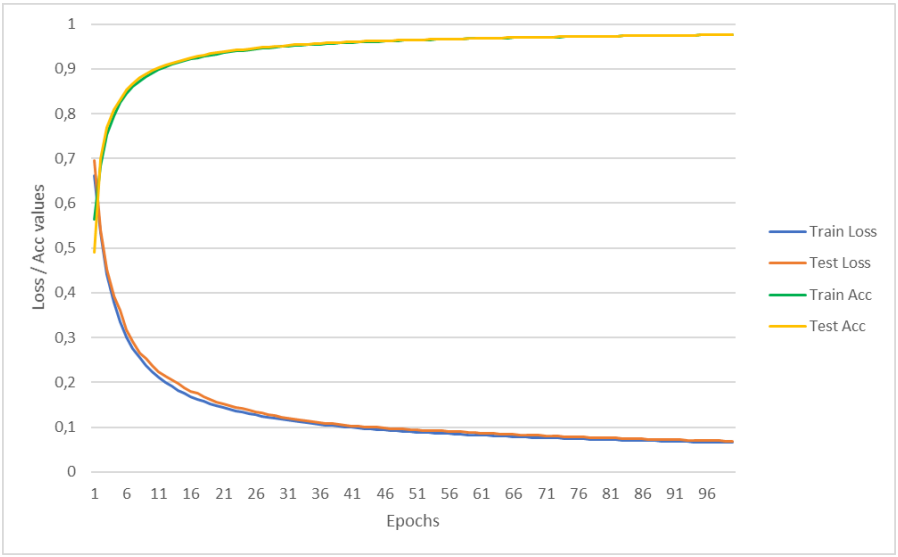


Figure 6.44: 5F RCV: training and test accuracy/loss for the LSTM model on dataset 5s 64Hz 3s overlap with randomized channels

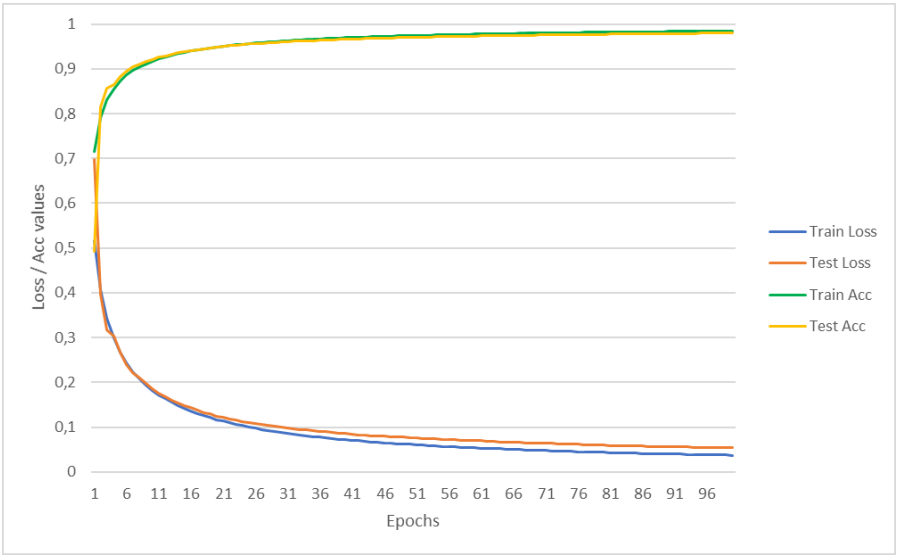


Figure 6.45: 5F RCV: training and test accuracy/loss for the CGRNN model on dataset 5s 64Hz 3s overlap with randomized channels

6.5.3 LOO

Metrics	LSTM	CGRNN	p-value
ACC	56.90 ± 7.95	56.20 ± 8.01	0.90
SEN	64.07 ± 11.06	57.30 ± 13.84	0.45
SPE	49.68 ± 10.92	55.16 ± 12.75	0.52

Table 6.54: Dataset 5s 64Hz 3s overlap: summary result of the LOO procedure on the test set

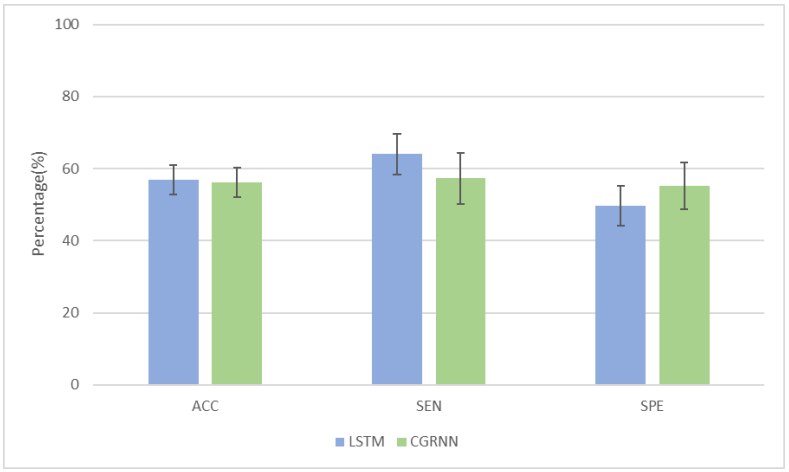


Figure 6.46: LOO: comparison between LSTM and CGRNN on dataset 5s 64Hz 3s overlap. The data are the one in table 6.54

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	1797	6703	23,19	14,64	31,79
Patient 2	898	13187	56,08	99,11	12,86
Patient 3	1798	13576	49,75	78,98	20,32
Patient 4	3598	22986	40,33	53,42	27,14
Patient 5	898	4343	59,04	45,21	72,93
Patient 6	6908	42659	80,39	68,56	92,26
Patient 7	898	4319	37,72	60,80	14,54
Patient 8	5398	27678	77,64	82,92	72,35
Patient 9	1798	6912	55,52	97,55	13,21
Patient 10	1153	7514	69,18	96,44	41,88
Patient 11	1798	8986	96,12	99,67	92,55
Patient 12	2698	35588	46,11	91,85	0,04
Patient 13	1798	5795	77,09	100,00	54,03
Patient 14	1348	4967	71,69	46,51	97,01
Patient 15	2025	9013	35,02	34,86	35,18
Patient 16	1169	9359	30,99	1,45	60,58
Patient 17	1798	8239	52,06	20,13	84,21
Patient 18	6067	47170	69,95	73,91	66,00
Patient 19	2698	14024	73,07	76,69	69,42
Patient 20	1798	16289	55,86	51,84	59,91
Patient 21	4498	41827	96,86	99,47	94,26
Patient 22	3598	17472	56,86	77,96	35,60
Patient 23	2088	15765	76,63	79,98	73,26
Patient 24	1475	9824	63,72	70,31	57,11
Patient 25	1798	7988	57,25	82,87	31,47
Patient 26	2698	21580	78,63	75,72	81,55
Patient 27	1798	14074	11,02	0,50	21,61
Patient 28	2698	19789	28,44	56,08	0,60
Patient 29	2978	9152	23,89	20,69	27,10

Table 6.55: Dataset 5s 64Hz 3s overlap on test set: results for the LOO procedure with LSTM divided by patient

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	1797	6703	16,60	29,60	3,53
Patient 2	898	13187	50,89	97,22	4,36
Patient 3	1798	13576	60,32	91,71	28,72
Patient 4	3598	22986	42,42	8,25	76,86
Patient 5	898	4343	56,64	20,04	93,40
Patient 6	6908	42659	94,28	97,26	91,25
Patient 7	898	4319	44,81	65,59	23,94
Patient 8	5398	27678	27,43	42,44	12,29
Patient 9	1798	6912	89,23	82,81	95,69
Patient 10	1153	7514	61,02	85,43	36,58
Patient 11	1798	8986	95,42	98,00	92,83
Patient 12	2698	35588	51,17	99,04	2,95
Patient 13	1798	5795	54,16	100,00	8,01
Patient 14	1348	4967	24,07	21,96	26,19
Patient 15	2025	9013	48,97	0,00	100,00
Patient 16	1169	9359	33,77	21,47	46,43
Patient 17	1798	8239	82,95	71,97	94,01
Patient 18	6067	47170	52,11	24,16	80,54
Patient 19	2698	14024	43,04	13,16	73,15
Patient 20	1798	16289	49,83	0,00	100,00
Patient 21	4498	41827	80,86	97,62	63,96
Patient 22	3598	17472	71,32	86,66	55,85
Patient 23	2088	15765	59,77	98,56	19,42
Patient 24	1475	9824	82,34	95,12	69,50
Patient 25	1798	7988	60,13	77,20	42,95
Patient 26	2698	21580	87,69	86,58	88,80
Patient 27	1798	14074	20,84	28,64	12,99
Patient 28	2698	19789	49,81	0,00	100,00
Patient 29	2978	9152	38,18	21,29	55,46

Table 6.56: Dataset 5s 64Hz 3s overlap on test set: results for the LOO procedure with CGRNN divided by patient

6.5.4 LOO randomized channels

Metrics	LSTM	CGRNN	p-value
ACC	58.10 ± 7.28	55.47 ± 7.77	0.63
SEN	66.53 ± 10.96	44.82 ± 13.79	0.19
SPE	49.65 ± 12.60	66.30 ± 12.91	0.076

Table 6.57: Dataset 5s 64Hz 3s overlap with randomized channels: summary result of the LOO procedure on the test set

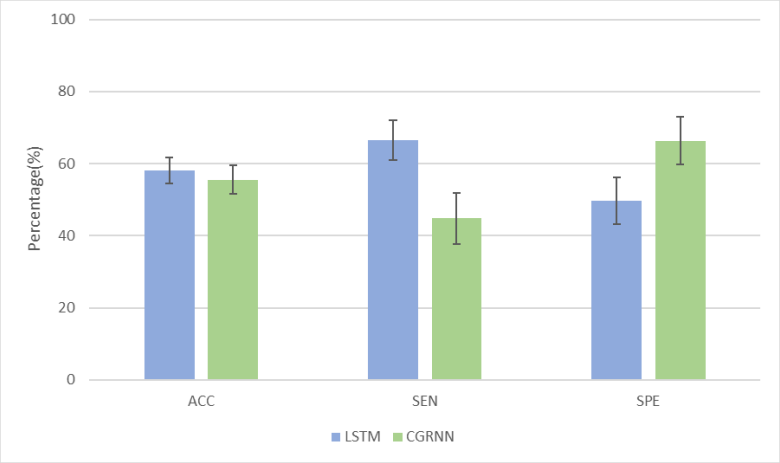


Figure 6.47: LOO: comparison between LSTM and CGRNN on dataset 5s 64Hz 3s overlap with randomized channels. The data are the one in table 6.57

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	1797	6703	16,60	29,60	3,53
Patient 2	898	13187	50,89	97,22	4,36
Patient 3	1798	13576	60,32	91,71	28,72
Patient 4	3598	22986	42,42	8,25	76,86
Patient 5	898	4343	56,64	20,04	93,40
Patient 6	6908	42659	94,28	97,26	91,25
Patient 7	898	4319	44,81	65,59	23,94
Patient 8	5398	27678	27,43	42,44	12,29
Patient 9	1798	6912	89,23	82,81	95,69
Patient 10	1153	7514	61,02	85,43	36,58
Patient 11	1798	8986	95,42	98,00	92,83
Patient 12	2698	35588	51,17	99,04	2,95
Patient 13	1798	5795	54,16	100,00	8,01
Patient 14	1348	4967	24,07	21,96	26,19
Patient 15	2025	9013	48,97	0,00	100,00
Patient 16	1169	9359	33,77	21,47	46,43
Patient 17	1798	8239	82,95	71,97	94,01
Patient 18	6067	47170	52,11	24,16	80,54
Patient 19	2698	14024	43,04	13,16	73,15
Patient 20	1798	16289	49,83	0,00	100,00
Patient 21	4498	41827	80,86	97,62	63,96
Patient 22	3598	17472	71,32	86,66	55,85
Patient 23	2088	15765	59,77	98,56	19,42
Patient 24	1475	9824	82,34	95,12	69,50
Patient 25	1798	7988	60,13	77,20	42,95
Patient 26	2698	21580	87,69	86,58	88,80
Patient 27	1798	14074	20,84	28,64	12,99
Patient 28	2698	19789	49,81	0,00	100,00
Patient 29	2978	9152	38,18	21,29	55,46

Table 6.58: Dataset 5s 64Hz 3s overlap on test set with randomized channels: results for the LOO procedure with LSTM divided by patient

Name	#Preictal	#Interictal	ACC (%)	SEN (%)	SPE (%)
Patient 1	1797	6703	13,98	24,99	2,91
Patient 2	898	13187	49,39	96,44	2,13
Patient 3	1798	13576	87,14	91,94	82,31
Patient 4	3598	22986	49,80	0,00	100,00
Patient 5	898	4343	57,14	35,41	78,97
Patient 6	6908	42659	64,60	32,46	97,30
Patient 7	898	4319	42,08	18,82	65,44
Patient 8	5398	27678	50,08	51,32	48,84
Patient 9	1798	6912	49,83	0,00	100,00
Patient 10	1153	7514	63,67	55,25	72,11
Patient 11	1798	8986	96,07	99,17	92,95
Patient 12	2698	35588	48,98	96,40	1,19
Patient 13	1798	5795	81,86	99,83	63,77
Patient 14	1348	4967	57,81	47,85	67,84
Patient 15	2025	9013	48,97	0,00	100,00
Patient 16	1169	9359	49,26	0,00	100,00
Patient 17	1798	8239	49,83	0,00	100,00
Patient 18	6067	47170	49,58	0,00	100,00
Patient 19	2698	14024	49,81	0,00	100,00
Patient 20	1798	16289	41,07	54,45	27,60
Patient 21	4498	41827	65,97	73,61	58,27
Patient 22	3598	17472	49,80	0,00	100,00
Patient 23	2088	15765	81,08	90,18	71,61
Patient 24	1475	9824	86,07	81,83	90,33
Patient 25	1798	7988	72,82	75,19	70,44
Patient 26	2698	21580	86,81	83,32	90,33
Patient 27	1798	14074	11,69	9,07	14,33
Patient 28	2698	19789	16,42	17,75	15,09
Patient 29	2978	9152	37,23	64,71	9,11

Table 6.59: Dataset 5s 64Hz 3s overlap on test set with randomized channels: results for the LOO procedure with CGRNN divided by patient

6.5.5 Filter feature selection

Variance		ACC(%)	SEN(%)	SPE(%)
1 channel	**LSTM**	71,86	73,31	70,44
	CGRNN	76,05	76,80	75,29
3 channels	**LSTM**	87,99	84,26	91,68
	CGRNN	88,25	90,90	85,57
5 channels	**LSTM**	93,26	94,51	92,04
	CGRNN	91,77	92,63	90,92
10 channels	**LSTM**	96,84	97,08	96,60
	CGRNN	95,31	96,07	94,56
15 channels	**LSTM**	97,76	99,34	96,18
	CGRNN	98,18	97,76	98,59
16 channels	**LSTM**	—	—	—
	CGRNN	98,77	98,46	99,06

Table 6.60: Dataset 5s 64Hz 3 second overlap: results for the feature selection procedure using the variance filter method.

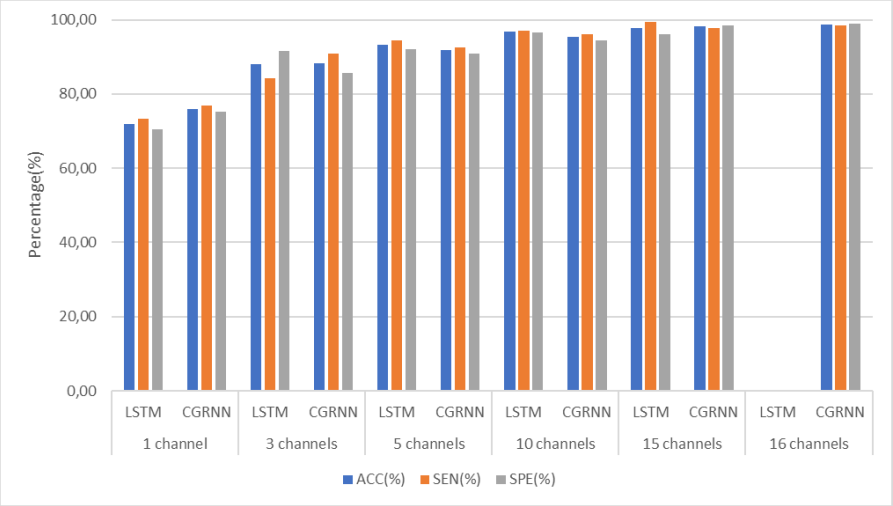


Figure 6.48: Dataset 5s 64Hz 3 second overlap: result for the variance feature selection based on the results on table 6.60

Variance Difference		ACC(%)	SEN(%)	SPE(%)
1 channel	**LSTM**	73,25	72,23	74,29
	CGRNN	76,65	78,72	74,55
3 channels	**LSTM**	87,08	82,86	91,23
	CGRNN	89,31	90,72	87,89
5 channels	**LSTM**	93,83	95,80	91,87
	CGRNN	94,57	94,55	94,59
10 channels	**LSTM**	95,90	93,66	98,14
	CGRNN	95,84	96,80	94,92
15 channels	**LSTM**	98,55	98,99	98,10
	CGRNN	98,19	98,06	98,31
16 channels	**LSTM**	—	—	—
	CGRNN	98,21	99,02	98,06

Table 6.61: Dataset 5s 64Hz 3 second overlap: results for the feature selection procedure using the variance difference filter method.

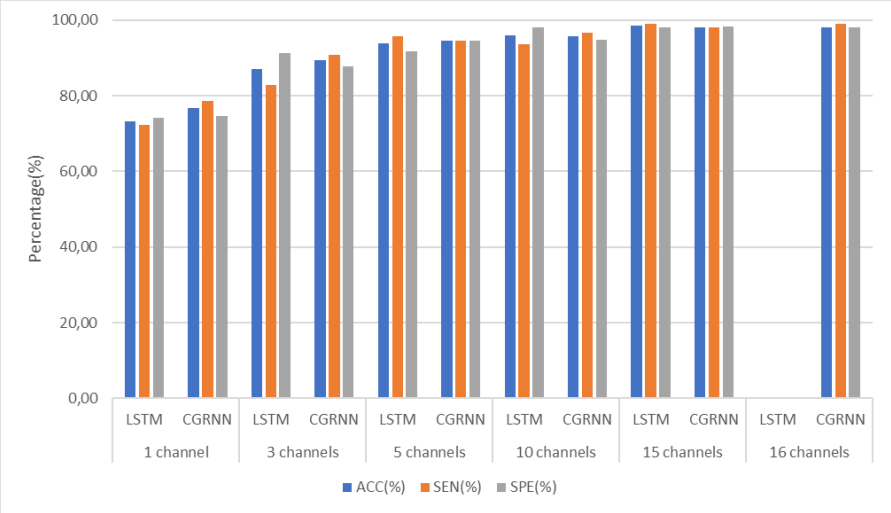


Figure 6.49: Dataset 5s 64Hz 3 second overlap: result for the variance difference feature selection based on the results on table 6.61

6.5.6 Wrapper feature selection

Metrics	LSTM	CGRNN	p-value
ACC	97.47 ± 0.42	97.92 ± 0.44	0.18
SEN	97.35 ± 1.61	97.69 ± 0.53	0.70
SPE	97.55 ± 1.91	98.16 ± 0.82	0.60

Table 6.62: Dataset 5s 64Hz 3 second overlap: summary result of the wrapper feature selection method presented in listing 4.1

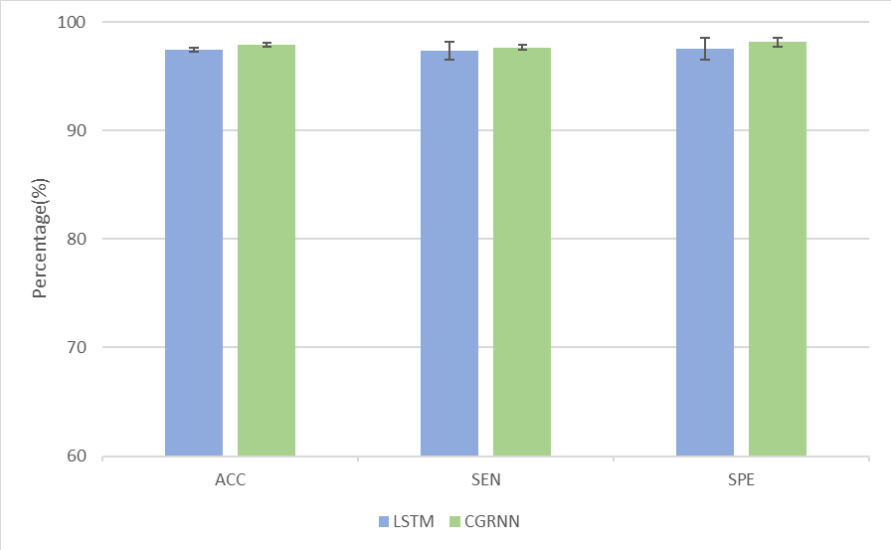


Figure 6.50: Dataset 5s 64Hz 3 second overlap: result for the wrapper feature selection method on the results on table 6.62

Method	ACC(%)	SEN(%)	SPE(%)
Variance	97,76	99,34	96,19
Var-Diff	98,55	98,99	98,10
Wrapper	97,47	97,35	97,57

Table 6.63: Dataset 5s 64Hz 3 second overlap: comparison between the result of the three feature selection methods on LSTM with 15 channels

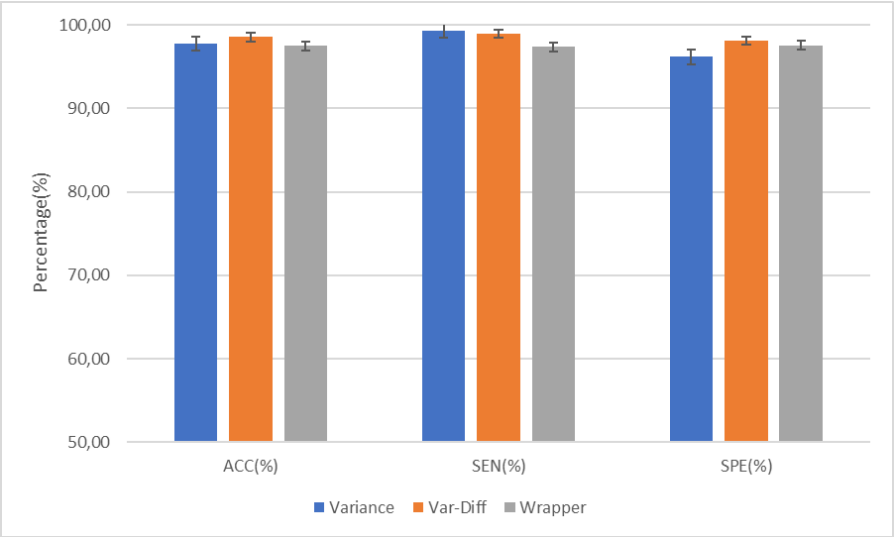


Figure 6.51: Dataset 5s 64Hz 3s overlap: comparison between the three feature selection methods on LSTM using the results in table 6.63

Method	p-value
Wrapper-Variance	0,72
Wrapper-VarDiff	0,34
Variance-VarDiff	0,16

Table 6.64: Dataset 5s 64Hz 3s overlap: p-value between the various feature selection methods for LSTM on accuracy metric

Method	ACC(%)	SEN(%)	SPE(%)
Variance	98,77	98,46	99,06
Var-Diff	98,21	99,02	98,06
Wrapper	97,92	97,69	98,16

Table 6.65: Dataset 5s 64Hz 3 second overlap: comparison between the result of the three feature selection methods on CGRNN with 15 channels

Method	p-value
Wrapper-Variance	0,12
Wrapper-VarDiff	0,39
Variance-VarDiff	0,52

Table 6.66: Dataset 5s 64Hz 3s overlap: p-value between the various feature selection methods for CGRNN on accuracy metric

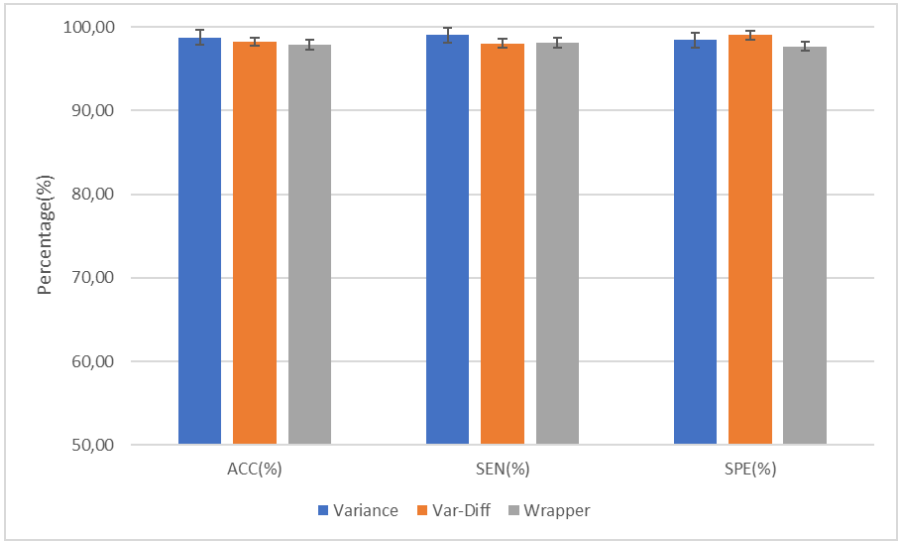


Figure 6.52: Dataset 5s 64Hz 3s overlap: comparison between the three feature selection methods on CGRNN using the results in table 6.65

6.6 Discussion

In this study, we used raw signals from multichannel EEG recordings from 29 patients with epilepsy and conducted a comparative analysis between a model that uses only LSTM and a model that is the union of 1D CNN and GRU. The performance evaluation methods were RCV and LOO, using both registered channels and randomized channels.
In addition to this, a study was also conducted on the actual informative capacity of each channel, using feature selection techniques to eliminate the least informative channels for solving the prediction task.
However, these results cannot be compared with other similar works because the dataset used for the analyses is not available online because it was collected from a local hospital.

The results obtained from the RCV validation (Tables: 6.2, 6.20, 6.36 and 6.52) suggest that both models are able to achieve excellent performance in seizure prediction with an accuracy of 98.45% for LSTM and 96.9% for CGRNN. This is in line with expectations. The main difference between LSTM and GRU is that the latter are much faster to train at the expense of a slight decrease in performance. Based on the tests, CGRNN takes 5 seconds per epoch of training in the case of a data set without overlap and 15 seconds with overlap. On the other hand, the LSTM model takes 16 seconds in the case without overlap and 42 seconds in the case with overlap. Therefore, it should be noted that the CGRNN is significantly faster (about three times) than the LSTM at the expense of a decrease in accuracy of 1.55%.
Both models result in very good sensitivity and specificity performance. The LSTM results in a sensitivity of 98.5% and a specificity of 98.48%, while the CGRNN results in a sensitivity of 98.3% and a specificity of 98.0%. The results of the p-value values in all tables are significantly higher than the 5% threshold, showing that these fluctuations in performance should not be considered to be the fault of the models' internal architecture and the features they extract, but only due to random fluctuations in training.
In the graphs 6.2, 6.15, 6.28 and 6.41 for LSTM and graphs 6.3, 6.16, 6.29 and 6.42 for CGRNN, the blue line represents the training loss, while the orange line represents the test loss. These two, as the number of epochs increases, continuously decrease. This shows that the models are able to learn features that allow the task to be solved without overfitting. This behavior was expected, as the parameters were optimized using Optuna on the test performances. Further evidence that the models have learned salient features are the green and yellow lines that represent training and test accuracy. These, as the number of epochs increases, also increase continuously, demonstrating how the network is able not only to learn but also to generalize to unseen data, thus improving its predictive capabilities.
One thing to note is that the CGRNN graphs show more continuous test losses compared to the LSTM graphs, which result in momentary peaks that then settle as the training increases. This indicates that CGRNN is able to learn more robust

features and have more reliable training. This can also be seen from the confidence intervals, which, in the case of the CGRNN, are more contained than those of the LSTM.

Very similar considerations can also be made in the case of randomized channels. The results reported in Tables 6.3, 6.21, 6.37 and 6.53 and the training curves that we can find in Figures 6.5, 6.18, 6.31 and 6.44 for LSTM and in Figures 6.6, 6.19, 6.32 and 6.45 for CGRNN show how the models behave similarly in the case of using channels in recording order and in random order. This similarity is due to the fact that the models are designed to take all channels as input at once and use them together. This was done to give the model freedom to exploit the interchannel correlation in the best possible way. This allows the models to have more freedom during the training phase, allowing them to learn more robust features that are independent of the order.

Unfortunately, these considerations do not hold when moving to a more robust evaluation setup like LOO.
As it is assumed that the models are able to learn significant features for the task, in Tables 6.4, 6.22, 6.38 and 6.54 we can see that all three metrics have a very significant drop in performance. For the LSTM model, accuracy, sensitivity and specificity go from 98.45%, 98.5% and 98.48% to 60.0%, 64.3% and 55.8%, with a drop of more than 30 percentage points. A similar situation occurs in the case of CGRNNs, where accuracy, sensitivity, and specificity go from 96.9%, 98.3% and 98.0% to 55.6%, 51.6% and 59.5%.
As can be seen in the tables divided by patients (Tables 6.5, 6.23, 6.39 and 6.55), we have some patients for whom the accuracy, sensitivity, and specificity values are good (example Patient 23: 87.14% - 94.49% - 79.0% and Patient 6: 86.90% - 78.32% - 95.55%) while others have really low performances (example Patient 28: 16.41% - 24.63% - 7.23% and Patient 27: 11.02% - 0.50% - 21.61%).
This can be associated with many factors, such as the fact that the dataset is not homogeneous in terms of types of epilepsy. This means that some patients have similar pathologies and the models can predict their seizures even if the patient has never been seen in the training phase, while for other patients, with different types of epilepsy, this does not happen.
This shows that it is difficult to develop models that can predict epileptic seizures regardless of the type of epilepsy, because the different types of epileptic seizures are probably characterized by totally different temporal features. It is also possible that there are no common patterns across all types of epilepsy that would allow a model to predict them correctly regardless of the type. One thing worth noting is that the results are more or less consistent between the various tests performed.
A patient who has low or high performance in one dataset will also have those performances in all other datasets. For example, Patient 1 has low performance in all datasets, with accuracy, sensitivity, and specificity values of 23.9%, 13.8% and 62.9%. Patient 21, on the other hand, has much better performance, with values that are around 86.7%, 99.4% and 73.4%.
This observation could lead to further investigation of patient data to understand

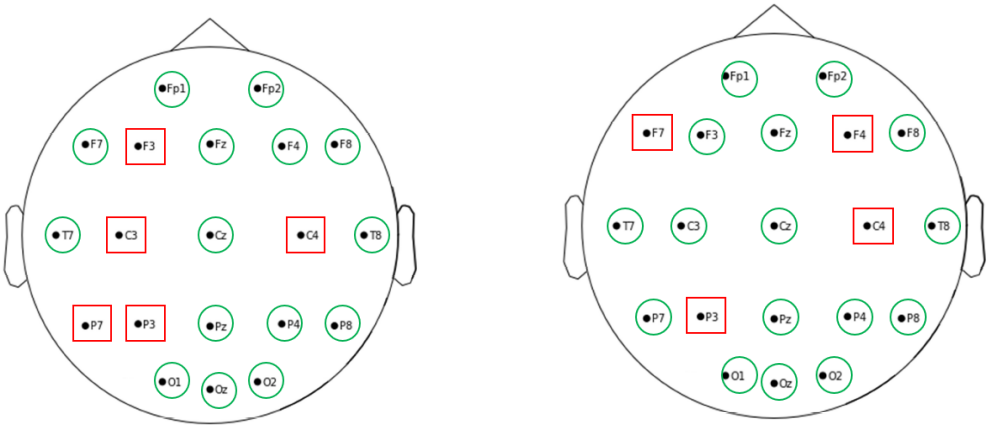
the causes of these differences. Possible explanations include the type of epilepsy, the quality of the recorded data, or the lack of sufficient data for each type of epilepsy.
The same considerations can be made in the case of CGRNNs, the results of which can be found in Tables 6.6, 6.24, 6.40 and 6.56, as well as in the case of randomized channels (Tables 6.8, 6.26, 6.42 and 6.58 for LSTM and Tables 6.9, 6.27, 6.43 and 6.59 for CGRNN).

The last series of experiments that were carried out concerned feature selection, that is, the selection of the most informative channels for prediction. Starting from the 20 total channels (Table 4.1), it was attempted to reduce them using two techniques: filtering and wrapper.
Tables 6.11, 6.28, 6.44 and 6.60 show the results of the variance-based filtering method. What can be seen is that as the number of channels increases, regardless of the model used, the performances increase. This shows that there are no totally irrelevant channels for seizure prediction, but that they all contain relevant information. This also applies to the case of the results for the variance difference method in Tables 6.12, 6.29, 6.45 and 6.61.
As already seen in Table 6.10, the two methods classify the channels differently in terms of informativeness. However, the results still show an improvement in performance as the number of channels increases, even if the subset of channels used is different. This suggests that the channels, although different, share redundant information. If this information is not available from one channel because it is excluded from the training, it can be retrieved from another channel. This makes the training robust to the presence of noise or lack of data in a specific channel.

As for the wrapper methods, we instead have that, depending on the model used, the subset of channels is different (Table 6.13). In Figure 6.53 we can find a representation of the selected channels (green) and the unselected channels (red) from the wrapper method applied to the LSTM model **(a)** and the CGRNN model **(b)**.
The comparison of the features selected by the LSTM and the CGRNN (Tables 6.14, 6.30, 6.46 and 6.62) shows a behavior similar to what was already seen for the 5-fold RCV procedure.
The LSTM model has better results than the CGRNN even though it selected a smaller number of channels. However, the p-value analysis shows that, for each performance metric, the values are above the 5% threshold, thus showing no significance in the performance differences between the two models.

Tables 6.18, 6.34, 6.50 and 6.66 provide a direct comparison between the various feature selection methods. These tables report the p-values, in terms of accuracy, and show that all methods, compared to each other, have values greater than 5%. The analysis showed that all 20 channels are informative, even if not to the same extent. Some channels provide information that other channels cannot provide, and this redundant information is used to compensate for the lack of information from

other channels. This conclusion is supported by the fact that the performance is good in each proposed method, even if the subset of channels used is different for each model and method, including different areas of the brain (Fig. 6.53). Furthermore, p-values do not show significance in the diversity of reported performances.



(a) LSTM 15 channels selected by wrapper method **(b)** CGRNN 16 channels selected by wrapper method

Figure 6.53: EEG scalp electrode position. In green the channel used and red the ones not selected by the wrapper method

7 | Conclusions

Epilepsy is a neurological disease characterized by the occurrence of seizures. These are defined as *sudden and recurrent events that cause sensory and motor alteration, absence states, and loss of consciousness*. Seizures are an expression of altered electrical activity in brain neurons that can be caused by:

- **genetic causes**: inherited from parents;
- **acquired causes**: due to trauma, infections, or brain tumors.

Epilepsy is a disease that can cause serious complications in the patient's life, since

- it can cause falls that lead to physical injuries;
- it can limit social participation by preventing normal activities such as driving or working;
- it can cause anxiety, depression, and isolation;
- in extreme cases, it can lead to the patient's death.

Epilepsy can be treated with antiepileptic drugs, but some patients suffer from drug-resistant epilepsy (AED) that cannot be treated with conventional methods. The research described in this paper aims to analyze and deepen the deep learning techniques that, through EEG, can help to monitor brain activity and potentially predict the onset of seizures, to allow patients to avoid dangerous situations and plan preventive treatments.

In this paper, two artificial intelligence models based on recurrent neural networks are described: LSTM and CGRNN.
The LSTM model fully exploits the potential of LSTM layers, which are a particular type of recurrent neural network that can learn temporal dependencies even over long timesteps thanks to their internal structure.
The CGRNN model, on the other hand, is a combination of 1D convolutional neural networks that allows the automatic extraction of features from raw EEG signals and GRUs, which are a simplified version of LSTMs for learning temporal dependencies within data.

The experiments conducted show that less strict evaluation techniques (5-fold RCV) lead to excellent results in the prediction of epileptic seizures, providing a foundation for more robust evaluations and demonstrating that the models are potentially capable of discovering patterns within brain signals.
The real challenge is to create models that can achieve acceptable performance even when evaluated with more robust methods, such as LOO.
This technique is particularly important because models that can achieve good performance in this validation technique would be able to predict seizures in patients never seen before, thus improving the lives of these people.
Unfortunately, this is a very complex task due to numerous factors that make it very challenging to identify patterns for the preventive detection of epileptic seizures. Some of these factors are:

- psychophysical characteristics of patients;
- EEG recordings highly sensitive to any fluctuation and/or noise;
- different types of epilepsy are characterized by EEG patterns that vary significantly among them.

In this study, EEG recordings from 29 patients with epileptic seizures were used. From these recordings, four different datasets were derived on the basis of the length of the signal and the amount of information contained. In total, 56 experiments were conducted to demonstrate the actual ability of the models to predict epileptic seizures.
In all experiments involving 5-fold cross-validation, the models achieved excellent results that can serve as a basis for further evaluations.
Regarding experiments using the leave-one-out (LOO) validation technique, the models did not achieve overall good results. However, what can be concluded from these experiments is that there is still a significant margin for improvement because, for some patients, the models were able to predict the onset of epileptic seizures quite accurately. This suggests that common patterns have been learned and can be used to generalize to other patients.
The feature selection experiments, which aim to understand which channels are more important for the prediction task, have demonstrated that all channels carry relevant information. With the available data, it is not possible to reduce the number of channels with confidence. This opens up numerous potential analyses and future experiments.

What can be done is try to understand what distinguishes the patients for whom the LOO validation performed well compared to those for whom it did not, and try to create homogeneous datasets based on common epilepsy characteristics. It may be the case that there are no common patterns across all types of epilepsy and that ad hoc datasets and models are needed.
This could involve a more in-depth analysis of which channels activate anomalously and which do not. By selecting the features of each individual patient, it may be possible to discover which channels are more active. Consequently, grouping patients

on the basis of common channels could lead to the creation of more homogeneous datasets. This approach might allow models to better generalize to specific types of seizures.

Another possibility is to create model adaptation procedures. The models could be trained on a heterogeneous set of patients and then, for each patient, fine tunning could be applied to their specific data.
This would allow basic models to specialize in specific types of seizures, significantly improving their performance. Since they are already trained, the adaptation process through fine-tuning would require less time compared to training from scratch. This technique is known as Leave-One-Patient-Out with adaptation, which has been studied but not implemented in this document.

Last but not least, in support of all the other experiments, it would be to implement interpretability methods that can help researchers better understand why those outputs are generated and which features are extracted and used within the models. This is done because it is fundamental, especially in the medical field, to implement interpretable artificial intelligence methods. Explainable AI (XAI) enhances the reliability and transparency of medical diagnoses and AI-based treatment recommendations, fostering trust among healthcare professionals and patients.

Acronyms and Abbreviations

AED Anti-Epileptic Drug. 9, 119, 126

AI Artificial Intelligence. 10

BPTT Backpropagation through time. 13, 127, 128

CGRNN Convolutional Gated Recurrent Neural Network. 39

CNN Convolutional neural netwrok. 2, 11, 20

Conv Layer Convolutional layer. 20

DBS Deep brain stimulation. 10

EEG Electroencephalogram. 1, 10, 25, 126

FFN Feed Forward Network. 11, 126

GRU Gated recurrent unit. 2, 11, 15

LOO Leave One patient Out. 30, 120, 127

LSTM Long short-term memory. 2, 11, 15

RCV Randomized Cross Validation. 30, 127

RNN Recurrent neural network. 1, 11, 125

RNS Responsive neurostimulation. 10

TBPTT Trucated backpropagation through time. 15, 128

VNS Vagus nerve stimulation. 10

XAI Explainable AI. 121, 126

Glossary

Aura set of physical, mental, or autonomic sensations that a patient feels before a seizure. 6

Backpropagation through time gradient-based technique used to train recurrent neural networks (RNNs). The algorithm works by unrolling all input in timesteps. Each timestep is like a separate layer in a deep network, and the weights are updated based on the error at that timestep. The errors are propagated backwards through the network. This allows the network to learn from historical data and make more accurate predictions. 13, 127

Backpropagation Backpropagation is a supervised learning algorithm used to train artificial neural networks. It works by calculating the gradient of the loss function with respect to the network's weights and then using this gradient to update the weights in a way that reduces the loss. It works by repeatedly propagating the error signal backward through the network, from the output layer to the input layer. In each layer, the error is used to calculate the gradient of the loss function with respect to the weights of that layer. The weights are then updated in a way that reduces the gradient. 13

Bayesian Optimization Search search algorithm that uses a probabilistic model to guide the search process. It can be used to find the best hyperparameter values efficiently in problems with a large number of hyperparameters or where the objective function is expensive to evaluate. 39

Convolutional neural network type of deep learning architecture primarily used for processing grid-like data, such as images and videos. It excels at automatically extracting features from these data types through the use of convolutional layers. 2, 20

Deep brain stimulation type of neuromodulation approach. Electrodes are implanted in the brain, in the area that is believed to be responsible for seizures. Electrodes deliver electrical pulses to the brain at regular intervals. 10

Dropout regularization technique that randomly sets a fraction of the input units to zero during training. This forces the model to learn to rely on a more diverse set of features, which can help to prevent overfitting. 40

Drug resistent epilepsy refers to a condition in which seizures in patients with epilepsy are not adequately controlled despite treatment with multiple antiepileptic drugs (Anti-Epileptic Drug (AED)s). 9

Electroencephalogram EEG is a test that measures the electrical activity in your brain. It is a noninvasive procedure that uses small metal discs (electrodes) that are attached to your scalp. The electrodes detect the tiny electrical charges that result from the activity of your brain cells. It is the result of Electroencephalography. 1, 25

Electroencephalography method to record an electrogram of the spontaneous electrical activity of the brain. 126

Epilepsy chronic noncommunicable disease of the brain coused by seizure episodes that are the result of excessive electrical discharges in a group of brain cells that can take place in different parts of the brain. 1, 128

Explainable AI XAI also known as interpretable AI is a field of AI that focuses on making models and decision-making processes more transparent and understandable to humans. It aims to provide information on how AI models arrive at their decisions, allowing users to comprehend the reasoning behind those decisions, identify potential biases, and assess the reliability of the model's output. 121

Feature selection process of identifying and choosing a subset of the most relevant features or attributes from the input data. This is done with the aim of reducing the complexity of the model, improving its performance, and potentially speeding up training. 2, 30

Feed forward network is a type of artificial neural network in which the connections between nodes do not form a cycle. This means that information flows in a single direction, from the input layer to the output layer, without any feedback loops. Feedforward neural networks (FFN) are the simplest type of neural network and can be used to solve a wide range of problems, including classification, regression and pattern recognition. 11

Fine Tunning is a technique that involves adjusting the parameters of a pretrained model to improve its performance on a new task or dataset. It is particularly useful for tasks where limited training data is available. 121

Focal seizure type of seizure that starts in one area of the brain. It is also known as a partial seizure. Focal seizures can cause a variety of symptoms, depending on the area of the brain affected. 6

Gated Recurrent Unit type of RNN designed for processing sequential data. It shares similarities with LSTM but uses a more streamlined structure, making it computationally more efficient while still being effective in capturing long-term dependencies. 2, 15

Generalized seizure type of seizure that affects both sides of the brain at the same time. 6

Grid search method that exhaustively evaluates all possible combinations of hyperparameter values. This means that grid search is more likely to find the optimal set of hyperparameters, but it is also more computationally expensive than random search. 38, 127

Leave One patient Out LOO is a type of cross-validation where each patient in the dataset is used as a test set once, and the remaining patients are used as the training set. This process is repeated for all patients in the dataset, and the average performance across all test sets is used as an estimate of the model generalization error. 30

Lennox-Gastaut syndrome type of childhood epilepsy that is particularly severe. It almost always starts before age 10, with a diagnosis most likely to occur between ages 3 and 5. This condition causes multiple types of seizures that can cause permanent brain damage resulting in learning difficulties and other disabilities [4]. 8

Long Short-Term Mermory type of RNN that excels in capturing long-term dependencies in sequential data. It achieves this by using specialized memory cells that retain and retrieve information over extended sequences. 2, 15

Random search method for hyperparameter optimization that selects hyperparameter values randomly from a predefined range. This means that random search can explore a wider range of hyperparameter values than grid search, but it is also less likely to find the optimal set of hyperparameters. 38, 127

Randomized Cross Validation RCV is an evaluation technique that works by randomly splitting the dataset into k folds, and then training the model on k-1 folds and testing it on the remaining fold. This process is repeated k times, and the average performance across all k folds is used as an estimate of the model's generalization error. 30

Recurrent neural network is a type of artificial neural network which uses sequential data or time series data. They are distinguished by the concept of memory, since they take information from prior inputs to influence the current input and output. While traditional deep neural networks assume that input and output are independent of each other, the output of recurrent neural networks depends on the prior elements within the sequence. They leverage on backpropagation through time (BPTT). 1

Responsive neurostimulation type of neuromodulation approach. Electrodes are implanted in the brain, in the area that is believed to be responsible for seizures. The electrodes monitor brain activity and deliver electrical pulses to the brain when seizures are detected. 10

Seizure unstable situation in epilepsy patients due to excessive electrical discharge by brain cells. It is the cause of Epilepsy attacks. 1

Trucated backpropagation through time TBPTT is a modified version of the backpropagation through time BPTT algorithm that is used to train recurrent neural networks. BPTT is a standard algorithm for training RNNs, but it can be computationally expensive and memory intensive for long sequences. TBPTT addresses this problem by truncating the backpropagation process to a fixed number of timesteps. 15

Vagus nerve stimulation type of neuromodulation approach. A small device is implanted under the skin of the chest, connected to a wire that wraps around the vagus nerve. The device delivers electrical pulses to the nerve at regular intervals. 10

References

[1] Abdelhameed, M. Ahmed, and M. Bayoumi, "An efficient deep learning system for epileptic seizure prediction," pp. 1–5, 2021. DOI: 10.1109/ISCAS51556.2021.9401347 (cit. on p. 1).

[2] G. Alarcón and A. Valentin, *Introduction to Epilepsy*. Cambridge University Press, 2012. DOI: https://doi.org/10.1017/CBO9781139103992.

[3] R. G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger, "Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state," *Phys. Rev. E*, vol. 64, p. 061 907, 6 2001. DOI: 10.1103/PhysRevE.64.061907. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.64.061907 (cit. on p. 25).

[4] E. Bromfield, J. Cavazos, and J. Sirven. "An introduction to epilepsy." (), [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK2510/. (accessed: 05.10.2023) (cit. on pp. 5, 127).

[5] I. Cherifi. "Backpropagation through time explained." (), [Online]. Available: https://iq.opengenus.org/back-propagation-through-time/. (accessed: 18.10.2023) (cit. on p. 15).

[6] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," pp. 103–111, 2014. DOI: 10.3115/v1/W14-4012 (cit. on p. 15).

[7] B. Explained. "Intuitive guide to convolution." (), [Online]. Available: https://betterexplained.com/articles/intuitive-convolution/. (accessed: 25.10.2023) (cit. on p. 23).

[8] E. Foundation. "Epilepsy foundation." (), [Online]. Available: https://www.epilepsy.com/. (accessed: 09.10.2023) (cit. on p. 6).

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org (cit. on p. 12).

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997 (cit. on p. 15).

[11] R. Jana and I. Mukherjee, "Deep learning based efficient epileptic seizure prediction with eeg channel optimization," *Biomedical Signal Processing*

and Control, vol. 68, p. 102 767, 2021, ISSN: 1746-8094. DOI: https://doi.org/10.1016/j.bspc.2021.102767. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1746809421003645 (cit. on p. 33).

[12] Merck and Co. "Malattie-neurologiche: Epilessia." (), [Online]. Available: https://www.msdmanuals.com/it-it/professionale/malattie-neurologiche/epilessia/epilessia. (accessed: 05.10.2023) (cit. on pp. 5, 8).

[13] C. Olah. "Conv nets: A modular perspective." (), [Online]. Available: http://colah.github.io/posts/2014-07-Conv-Nets-Modular/. (accessed: 25.10.2023) (cit. on pp. 21, 22).

[14] C. Olah. "Understanding lstm networks." (), [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/. (accessed: 16.10.2023) (cit. on pp. 12, 14, 16–20).

[15] C. Olah. "Undestanding convolutions." (), [Online]. Available: http://colah.github.io/posts/2014-07-Understanding-Convolutions/. (accessed: 25.10.2023) (cit. on p. 23).

[16] Optuna. "Optimize your optimization." (), [Online]. Available: https://optuna.org/. (accessed: 30.10.2023) (cit. on p. 38).

[17] W. H. Organization. "Epilepsy." (), [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/epilepsy. (accessed: 22.09.2023) (cit. on pp. 1, 5).

[18] T. Otaiby, F. Abd El-Samie, S. Alshebeili, and I. Ahmad, "A review of channel selection algorithms for eeg signal processing," *EURASIP Journal on Advances in Signal Processing*, vol. 2015, Aug. 2015. DOI: 10.1186/s13634-015-0251-9 (cit. on p. 33).

[19] C. Ranjan, *Understanding Deep Learning Application in Rare Event Prediction*. Connaissance Publishing, 2020. DOI: 10.13140/RG.2.2.34297.49765.

[20] S. Roy, I. Kiral-Kornek, and S. Harrer, "Chrononet: A deep recurrent neural network for abnormal eeg identification," 2018. arXiv: 1802.00308 [eess.SP]. [Online]. Available: https://arxiv.org/abs/1802.00308 (cit. on pp. 43, 44).

[21] S. Sarmast, A. Abdullahi, and N. Jahan, "Current classification of seizures and epilepsies: Scope, limitations and recommendations for future action," 2020. DOI: 10.7759/cureus.10549 (cit. on p. 5).

[22] D. Schmidt and W. Löscher, "Drug resistance in epilepsy: Putative neurobiologic and clinical mechanisms," pp. 858–77, Jun. 2005. DOI: 10.1111/j.1528-1167.2005.54904.x (cit. on pp. 1, 9, 25).

[23] S. Shafiezadeh, G. M. Duma, G. Mento, *et al.*, "Methodological issues in evaluating machine learning models for eeg seizure prediction: Good cross-validation accuracy does not guarantee generalization to new patients," *Applied*

Sciences, vol. 13, no. 7, 2022, ISSN: 2076-3417. DOI: 10.3390/app13074262. [Online]. Available: https://www.mdpi.com/2076-3417/13/7/4262 (cit. on pp. 26, 27).

[24] A. Testolin, S. Shafiezadeh, and M. Pozza, "A comparison of recurrent and convolutional deep learning architectures for eeg seizure forecasting," 2024 under review.

[25] K. M. Tsiouris, V. C. Pezoulas, M. Zervakis, S. Konitsiotis, D. D. Koytsouris, and D. I. Fotiadis, "A long short-term memory deep learning network for the prediction of epileptic seizures using eeg signals," *Computers in Biology and Medicine*, vol. 99, pp. 24–37, 2018, ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2018.05.019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S001048251830132X (cit. on p. 40).

[26] Wikipedia. "Epilepsy." (), [Online]. Available: https://en.wikipedia.org/wiki/Epilepsy. (accessed: 22.09.2023) (cit. on p. 1).