Chaos, Solitons and Fractals 171 (2023) 113388

Contents lists available at ScienceDirect

Chaos, Solitons and Fractals

journal homepage: www.elsevier.com/locate/chaos

Analysis of heartbeat time series via machine learning for detection of illnesses

Sidney T. da Silva [a,d,*], Moacir F. de Godoy [b], Michele L. Gregório [b], Ricardo L. Viana [c], Antonio M. Batista [a,e]

[a] *Pós-Graduação em Ciências/Física, Universidade Estadual de Ponta Grossa, Ponta Grossa, PR, Brazil*
[b] *Faculdade de Medicina de São José do Rio Preto, São Joé do Rio Preto, SP, Brazil*
[c] *Departamento do Física, Universidade Federal do Paraná, Curitiba, PR, Brazil*
[d] *Departamento do Química, Universidade Federal do Paraná, Curitiba, PR, Brazil*
[e] *Departamento de Matemática e Estatística, Universidade Estadual de Ponta Grossa, Ponta Grossa, PR, Brazil*

ARTICLE INFO

Keywords:
Machine learning
Heartbeat
Confusion matrix
Permutation entropy
Autonomic nervous system

ABSTRACT

The heart, a component of the cardiovascular system, is responsible for pumping oxygenated and deoxygenated blood. It does not behave like a metronome and normally there is a variation in the duration of the intervals between each heartbeat, called Heart Rate Variability (HRV). In the presence of diseases or with the progression of aging, there is a reduction in HRV due to dysfunction of the autonomic nervous system. The objective of this work is to show, using machine learning techniques, that these techniques are able to relate directly the variability of the heart with the degree of the disease. Producing, as a practical result, the use of these techniques in the prediction of different types of diseases by only analyzing their time series. One of the first techniques used in our work is the unsupervised learning algorithm (t-Stochastic Neighbor Embedding). We show that this algorithm is able to differentiate the type and degree of the disease just by analyzing time series, we demonstrate that it is possible to design a neural network architecture capable of learning these characteristics, relating cardiac variability and the disease. In a complementary analysis, we check that cardiac variability can be directly related to permutation entropy, proving that the healthier an individual is, the more stochastic his cardiac time series is. We build a classification algorithm, using deep learning, from the confusion matrix and the ROC curve. This algorithm can be used as an entry point in diagnosing patients by measuring their HRV.

1. Introduction

The heart is an organ that sends blood throughout the circulatory system, carrying oxygen and important nutrients to the cells. The cardiovascular system also removes carbon dioxide and waste products away from the body [1]. The normal heart rhythm can be affected by cardiovascular diseases and depressive disorders [2]. An important indicator of diseases is the variability of the heart rate [3], that refers to the regulation of sinoatrial node. It permits to observe the heart's ability under regulatory inputs [4] and has been used to monitor clinical conditions [5]. Lower heart rate variability can be related to abnormal adaptation of the autonomic nervous system [6]. Meyerfeldt et al. [7] investigated heart rate variability before the onset of ventricular tachycardia in patients with an implantable cardioverter defibrillator. Marwan et al. [8] proposed measures of complexity to analyze heart rate variability data. Analysis of heart rate variability was used as a predictor of mortality in cardiovascular patients of intensive care unit [9]. Liu et al. [10], using recurrence quantification analysis (RQA) in the HRV data, classified the HRV data into pathology groups using a multilayer perceptron classifier.

First coined by Samuel in 1959 [11], machine learning (ML) is the study of computational algorithms which are designed to improve automatically through experience [12]. ML techniques have been widely applied to different field of science, for instance in solid-state materials [13], wildfire [14], and genetics [15]. It has been considered to predict different types of diseases, such as brain [16] and kidney diseases [17]. Recently, Herry et al. [18] used ML on heart rate variability to investigate children that were exposed prenatally to the Zika virus.

The ability of ML for cardiovascular disease prediction was reported by Krittanawong et al. [19]. Parthiban and Srivatsa [20] applied ML methods in diagnosing heart disease for diabetic patients. They showed the possibility of identifying heart disease vulnerability in a person

[*] Corresponding author at: Pós-Graduação em Ciências/Física, Universidade Estadual de Ponta Grossa, Ponta Grossa, PR, Brazil.
E-mail address: sidneyt.silva82@gmail.com (S.T. da Silva).

https://doi.org/10.1016/j.chaos.2023.113388
Received 14 December 2022; Received in revised form 17 March 2023; Accepted 18 March 2023
Available online 6 April 2023

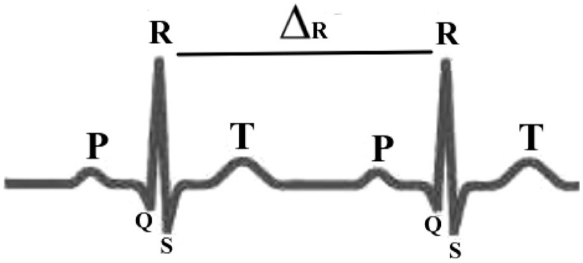

Fig. 1. The Δ_R interval series is the sequence of time intervals between successive beats on the ECG.

with diabetes. Recently, Ali et al. [21] demonstrated that ML algorithm has a very high accuracy and potential utility to make heart disease predictions.

In this work, we study how ML can be used to detect diseases by means of heartbeats. We analyze various experimental tachograms obtained from people with different clinical conditions. The experimental data are separated into healthy patients and patients with mild and serious illnesses. With regard to the ML techniques, we utilize a recurrent neural network coupled with a multilayer perceptron network. Dua et al. [22] demonstrated that multilayer perceptron method provides high classification accuracy of coronary artery disease utilizing heart rate variability analysis.

We apply the t-stochastic neighbor embedding (t-SNE) technique on our data sets. By means of t-SNE, we plot the 2D embedding visualization of different class targets related to diseases. We compute the confusion matrix to measure the performance of the machine learning classification. We also calculate the permutation entropy for the identification of diseases. Our results show that severe diseases exhibit higher permutation entropy difference values than mild diseases. The difference is between the entropies obtained from the cardiac and stochastic time series, both with the same temporal correlation quantifier.

This paper is organized as follows. In Section 2, we show the methodology about the machine learning. In Sections 3 and 4, we discuss our results about the analysis of empirical time series and permutation entropy, respectively. In the last section, we present our conclusions.

2. Methodology

2.1. Experimental data

The cardiac impulse travels through the heart and electrical currents travel through the surrounding tissues, consequently, a small proportion travels to the surface of the body. If the electrodes are placed on the skin, on opposite sides of the heart, the electrical potentials generated by these currents can be recorded. This record is known as an electrocardiogram (ECG). The normal ECG is composed of a P wave, a QRS complex, and a T wave (Fig. 1). The QRS complex is formed by three distinct waves: the Q wave, the R wave, and the S wave. The P wave is produced by the electrical potentials that are generated by the atria depolarization, before contracting. The QRS complex is due to the potentials that are generated when the ventricle depolarizes, before contracting. The T wave is due to the potentials generated during the recovery of the ventricles from the depolarized state. In summary, the electrocardiogram is composed of depolarization and repolarization waves.

The heart rate variability (HRV) signal consists of a series of time intervals between the R waves of the ECG, that is, the interval Δ_R. The series of intervals Δ_R is not equidistant in time. Our experimental data are obtained from HRV technique, which provides a powerful means for observing the interaction between the sympathetic and parasympathetic nervous system. The variation of the HRV can be used as an indicator of illness or early warning of impending heart disease. The analyses of the HRV are non-invasive and inexpensive tools to assess the health status of the circulatory system. Understanding the HRV brings us to the functional comprehension of the heart.

Fig. 2 shows the moving average of the normalized HRV signals for 5 groups of patients and a control group (healthy volunteers):

(1) Control group (Fig. 2(a)),
(2) Bipolar disorder patient group (Fig. 2(b)),
(3) Group of leprosy patients (Fig. 2(c)),
(4) Group of patients with chronic kidney disease (Fig. 2(d)),
(5) Group of brain dead patients (Fig. 2(e)),
(6) Intensive Care Units (ICU) patient group (Fig. 2(f)).

We normalize the groups across the moving average, for better visualization. In the learning algorithm, the dataset was normalized using variance and mean of the samples. For this normalization, we use sklearn's StandardScaler library.

The main objective of our work is to build a neural network capable of learning the dynamic characteristics of each group and to classify, through HRV signals, which group a patient belongs to. Using an unsupervised learning algorithm, we aim to group cardiac time series into groups with high correlation. This analysis is important to identify common characteristics without any supervision. We can use the information to extract common characteristics of these groups, improving the performance of the neural network at the time of classification. Another approach is the use of a methodology created by [23] that relates permutation entropy with the stochastically of a time series. This approach shows us how the variability of the HRV signal is directly related to the severity of the disease. And finally, we apply all this prior knowledge in creating a deep network, that is capable of classifying the degree of impairment of a patient by means of their cardiac time series.

2.2. Learning algorithm

We consider a neural network with the objective of classification. The network is trained to classify the time series into one of the groups. For the classification model, we use two coupled neural networks. A recurrent network with echo state (ESN) [24,25] coupled with a multilayer perceptron network (Fig. 3(a)) and using Principal Component Analysis (PCA) [26] as a dimension reducer.

The BDESN is utilized for the classification of time series $\mathbf{x} = \{\mathbf{x}_t\}_{t=0}^{T}$ labeled with class $\mathbf{c}$ through the following procedure. We first project the time series with smaller dimension $\mathbf{x}(t)$ to a larger space through the reservoir $(\mathbf{h}(t))$. Then a dimension reduction algorithm projects the reservoir outlet into a smaller space represented by the state vector $\mathbf{r_x}$, where all the dynamic characteristics of the input are represented by the vector $\mathbf{r_x}$. Finally, a multilayer perceptron (MLP) classifies the vector representative of $\mathbf{x}$. Fig. 3(a) details the procedure.

These state vectors with their reduced dimensions become the input vectors to the network (MLP), where the classification or regression takes place. The weights of these layers will undergo adjustments during training. At this point, a normal training of an MLP network is made.

In our architecture, we use a reading layer formed by an MLP network with three hidden layers of 400 neurons each one and an input layer with 500 neurons. On the hidden layers, we use a dropout = 0.2 (Figs. 3(a) and 3(b)) and "Greedy Layer-Wise" [27] as pre-training of the network. The optimizer and error function were "adam" and "MSE", respectively. In neural networks, an important rule is the choice of hyperparameters for a better performance in the classification model without suffering overfit. After choosing these hyperparameters, using Bayesian optimization (Appendix C), we apply our network for the classification problem. The best hyperparameters of the reservoir are: $N = 400$, $\alpha = 0.7607$, $\rho = 0.9698$, $\omega = 0.9$, $\eta = 0.0011$ and $PCA = 100$. Details of this procedure are described in Appendix A.

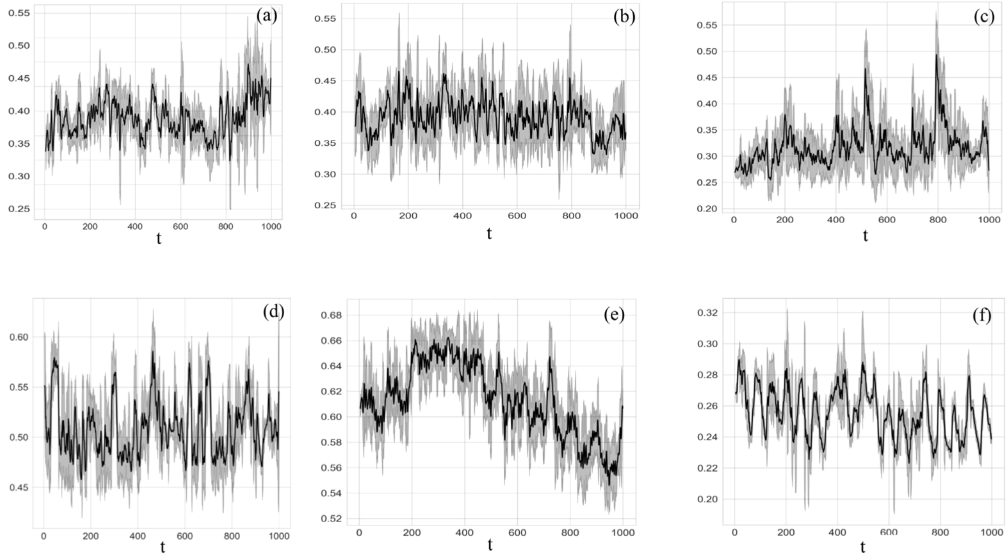

Fig. 2. Normalized HRV signal through the moving average for the six groups, where the full curve is the moving average and the light gray curve is the fluctuations. (a) Control group, (b) bipolar disorder patient group, (c) group of leprosy patients, (e) group of patients with chronic kidney disease, (d) group of brain dead patients, and (f) ICU patient group.

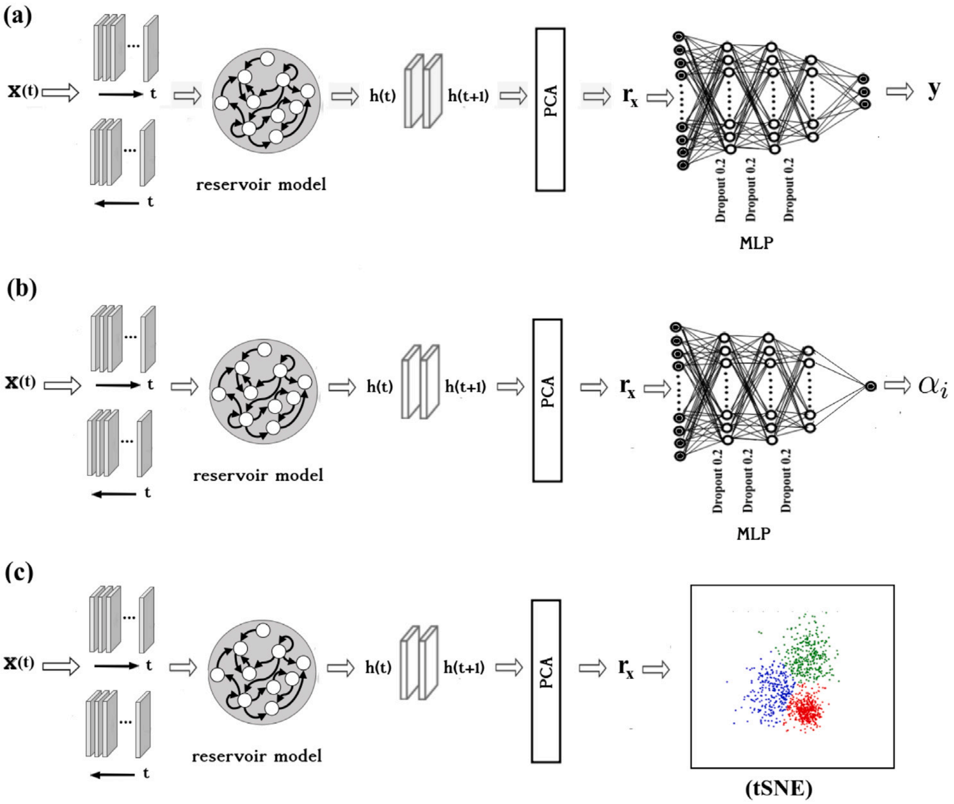

Fig. 3. Neural network. (a) It represents a deep network composed of a computational reservoir followed by a dimensional reduction (PCA) and a reading layer formed of a multilayer perceptron network trained for classification, (b) it represents a deep network composed of a computational reservoir followed by a dimensional reduction (PCA) and a reading layer formed of a multilayer perceptron network trained for regression and (c) it represents a deep network composed of a computational reservoir followed by a dimensional reduction (PCA) and a t-SNE layer, used for unsupervised learning.

2.3. Permutation entropy

i. Ordinal analysis and permutation entropy

Ordinal analysis allows the identification of patterns and nonlinear correlations in complex time series [28,29]. Each sequence of D data points in the time-series (consecutive or with a certain lag between them) is converted into a sequence of D relative values (smallest to largest) ordered from 0 to $D-1$, which defines an ordinal pattern. Then, the frequencies of occurrence of the different patterns in the time series define the set of ordinal probabilities, which in turn allows to calculate the information-theoretic measures, such as the permutation entropy. For instance, a sequence $\{0, 5, 10, 13\}$ in the time series transforms into the ordinal pattern "0123", while $\{0, 13, 5, 10\}$ transforms into "0312". As an example, Fig. 4 shows the ordinal patterns formed with D =

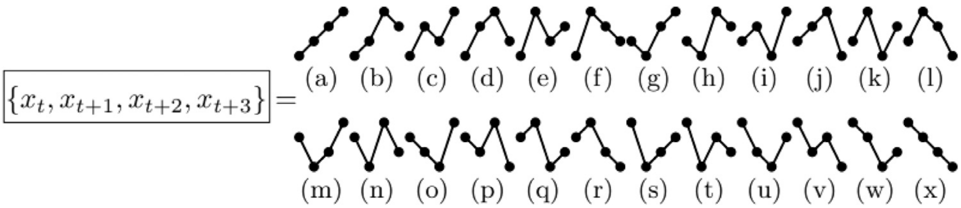

Fig. 4. Schematic illustration of 24 ordinal patterns that can be defined from D = 4 consecutive data values in a time series [23].

4 consecutive values. This sequence of time series points with size D is randomly selected. For more details on this procedure, read the work [30]. We evaluate the frequency of occurrence of each word, defined as the ordinal probability $p(i)$ with $\sum_{i=1}^{D!} p(i) = 1$, where i represents every possible word. Then, the permutation entropy is defined as the value of the Shannon entropy computed using all $p_i(i = 1, 2, ..D)$. Permutation entropy contains the information about the temporal structure associated with the underlying dynamics of a time series, therefore, permutation entropy seems to be particularly suited as a discriminative measure to reveal nonlinear dynamics in arbitrary real-world data [23].

$$S(D) = -\sum_{i=1}^{D} p(i) \ln p(i), \tag{1}$$

The permutation entropy varies from $S(D) = 0$ if the jth state $p(j) = 1$(while $p(i) = 0\ \forall, i \neq j$) to $S(D) = \ln D!$ if $p(i) = i/D!\ \forall i$. The normalized permutation entropy used in this work is given by:

$$\overline{S}(D) = \frac{S(D)}{\ln D!}. \tag{2}$$

To calculate the ordinal patterns, we use the algorithm proposed by Parlitz et al. [31]. We utilize $D = 6$ and no lag, i.e, the values of $D - 1$ overlap in defining two consecutive ordinal patterns. Therefore, we use the D! = 720 probabilities of the ordinal patterns. For a robust estimation of these probabilities, a time series of length $T \gg D!$ is needed. However, as shown in [23], the algorithm returns meaningful values even for time series that are much shorter.

3. Analysis of empirical time series

In our work, due to the difficulty in obtaining data from available volunteers, we worked with a total of 240 individuals, divided into 26 healthy individuals, 26 with leprosy, 38 with bipolar disorder, 26 with chronic kidney disease, 21 with brain death and 103 patients in the ICU. All time series obtained have a size of 1000 periods (Fig. 2). As we do not have a very large number of samples due to the difficulty of obtaining experimental data, we tried to compensate for this problem with a more sophisticated network and more efficient regularization, in order to be able to generalize our results and avoid overfitting.

Firstly, before employing our classification algorithm, we use the t-SNE algorithm (Appendix B) in order to pre-visualize and understand the correlations between the groups. Using this unsupervised learning algorithm, we group the cardiac time series into groups with high correlation. We process the data, without knowing a "prior" which class they belong to, letting our algorithm group them according to their common characteristics. In this work, we also use noisy time series with its α ranging from 0 to 2 ($P(f) = 1/f^{\alpha}$). These last artificial signals permit us to know how close the cardiac series are to them. Before using t-SNE in the data, we utilize a "BDESN" network (Fig. 3(c)). This network is fed with all time series and learns the dynamics of each series, generating as output the state vector $\mathbf{r}_X$, which brings all the information on the dynamics of the time series. With the unlabeled state variables, we feed our t-SNE algorithm. The result is shown in Fig. 5. The t-SNE groups data that exhibits a strong correlation and the ones closer to these "clusters" have similar dynamics. Noisy series have a very large variability, consequently, the cardiac series closer to these series present a greater variability. In Fig. 5(a), we see a proximity between the control group and patients with leprosy and bipolar disorder, who are practically included in the noisy series. On the other hand, patients with brain death and ICU are far from this region. It is possible to observe that patients with kidney problems move between these two groups. This algorithm shows us that patients with more severe illnesses have a lower variability compared with patients with less severe illnesses. Thus, for a better classification, we group these series into three large groups: Healthy individuals, individuals with mild illnesses and individuals with severe illnesses, where we include in this group patients with kidney diseases (Fig. 5(b)).

Table 1
Confusion matrix.

	Predicted positive	Predicted negative
Current positive	TP	FN
Current negative	FP	TN

By means of the t-SNE plot, we group the patients into three groups according to healthy, mild disease, and severe disease. This way, it is possible a better performance of the algorithm. We compute the performance measurement through the confusion matrix. The performance is evaluated based on three main measurement performances in RNN models, which are the accuracy, precision, and recall

$$\begin{aligned} Accuracy &= \frac{(TP + TN)}{(TP + TN + FP + FN)}, \\ Precision &= \frac{TP}{(TP + FP)}, \\ Recall &= \frac{TP}{TP + FN}, \\ F_1 score &= \frac{2 * Precision * Recall}{Precision + Recall}. \end{aligned} \tag{3}$$

These measurements are described using the confusion matrix that considers a two-class classification problem, as illustrated in Table 1. The main diagonal values are the correctly predicted values, while the off-diagonal values

In neural networks, an important rule is the choice of hyperparameters for a better performance in the classification model without suffering overfit. After choosing these hyperparameters, using Bayesian optimization, we can apply our network for the classification problem. The best hyperparameters of the reservoir are shown in Table 2. We focus on the HRV signal of a patient to classify it into healthy, medium disease or severe disease. Importantly, the diagnosis of our network is based on the variability of the HRV signal, therefore, its classification is independent of the disease. It compares the patient's signal, based on its variability, and classifies it in one of the three groups. According to the signal dynamics, the network is able to diagnose the severity of the patient. For the process of obtaining our algorithm, we first randomly separate 70% from the entire dataset for training and 30% for testing. This random separation must be done to maintain the representativeness of all classes, both in the training set and in the test set, guaranteeing a good generalization of these processes. We group the HRVs into three categories: group 0 (control), group 1 (leprosy and bipolar), and group 2 (chronic kidney disease, ICU and brain death).

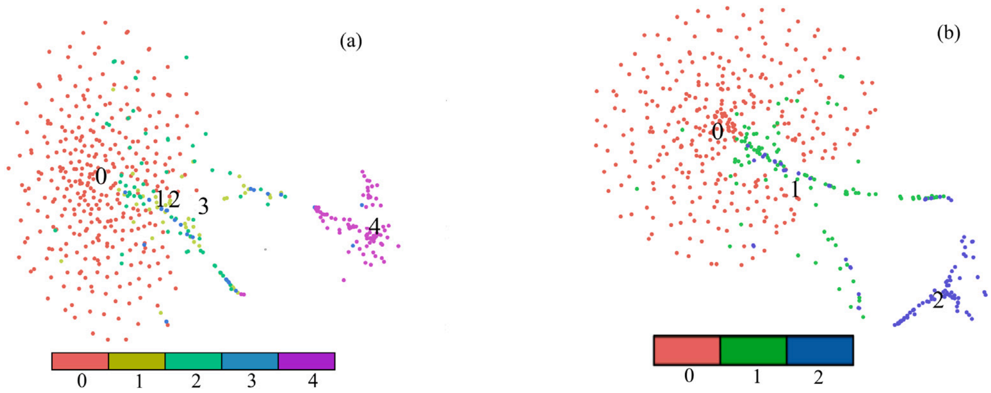

Fig. 5. T-SNE plot applied to noise (0), control (1), mild diseases (bipolar and leprosy) (2), Chronic Kidney Disease (3), and ICU and Death brain (4), as shown in panel (a). In the panel (b), t-SNE plot applied to healthy (noise and control) (0), mild diseases (bipolar and leprosy) (1), and severe diseases (IRC, UTI and Death brain) (2).

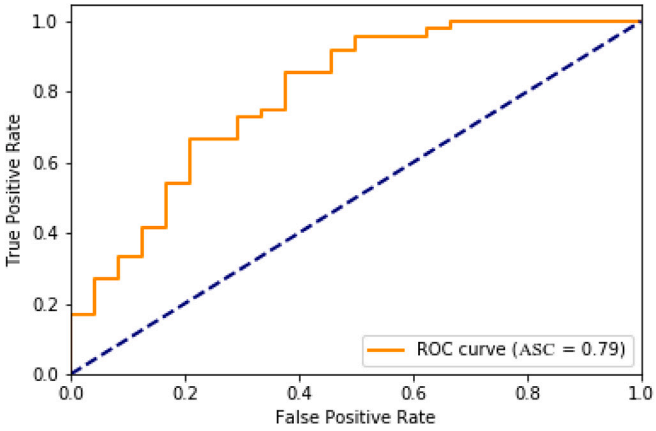

Fig. 6. The roc curve represents the accuracy performance of a perfect diagnostic test (ASC = 1.0) and a random error line (ASC = 0.5). Our results show ASC = 0.79.

This grouping is done through the level of disease severity. For the classification layer, the architecture of the MLP network (Fig. 3(a)) is made as follows: with four hidden layers with 500, 400, 400, and 400 neurons, respectively. For the regularization, we consider a dropout of 20% with an adaptive learning rate of 0.01 and with a minibatch of 32. In the output layer, we apply the softmax function, and for the gradient, we use *adam* [32]. The reservoir (BDESN), we utilize the hyperparameters (Table 2). After the training phase, the best accuracy is 80.85% in the test data with 80.03% f1 score. In Fig. 5, the confusion matrix of the test data is shown. The diagonal represents the hits, and the values outside the diagonal correspond to the erroneously classified data. The diagonals exhibit the recall and precision values for each class.

The relative shapes of the Receiver Operating Characteristic Curve (ROC curve) on the graph are a quick approach to estimate and compare the precision between tests (Fig. 6). A perfect test (ASC = 1) identifies all positive and negative results. An inaccurate test or similar to a coin toss would result in a 45 degree line (ASC = 0.5). ASC is the area swept out by the ROC curve. These two extremes (perfect test and uninformative test) are often used as references. ROC curves closer to a perfect test have a higher ASC and are more accurate than those closer to the random error line (ASC ≈ 0.5). In our results, we find ASC = 0.79.

We see that the highest precision is in the severe and control disease group, while the lowest precision is in the mild diseases. The severe illness has a dynamics different of the control group. This way, it is easier to separate the patients with severe disease and control than patients with mild disease, which has an intermediate dynamics. In Fig. 7, we observe a diagnostic accuracy of a cardiac time series greater than 80%, that is, if a patient is diagnosed in one of the 3 groups (healthy, mild, and severe), the probability of correctness is 80%. In other words, our neural network can be applied to the patient's first diagnosis with good precision. The medical diagnosis is performed directly on the time series without prior treatment of the collected data and without other more expensive diagnoses. Our neural network can be a gateway to a patient's diagnosis and depending on the group to start a more thorough investigation. This network is available at [33] or in the webapp [34].

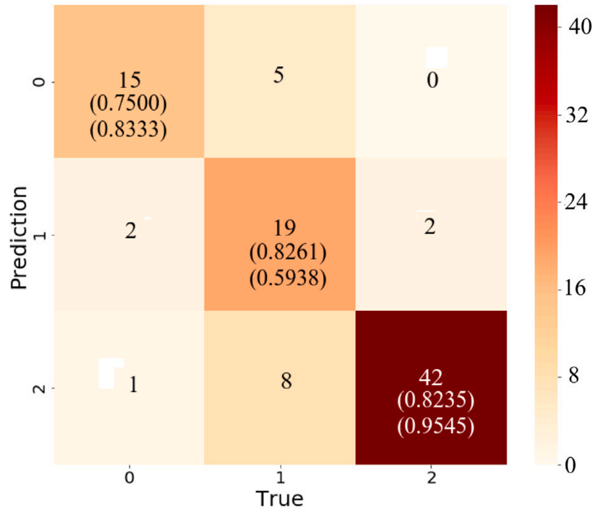

Fig. 7. Confusion matrix of test data. 0 — represents volunteers, 2 — mild diseases, and 2 — severe diseases. The top values represent precision and low values represent recall for each disease, respectively.

A way to characterize the complexity of a time series is that of determining the decay law of its power spectrum $P(f) = 1/f^{\alpha}$, where α is the correlation in the signal. We employ this complexity measure to determine the degree of variability of the time series, comparing its dynamics with a stochastic noise. We determine the value of α of some time series (cardiac series) and use it to generate a stochastic noise. Then, we compute the permutation entropy of each series and the generated noise, the greater the difference between the two entropies, the smaller the variability of the time series.

We train a recurrent neural network (Fig. 3(b)) with several time series (flicker noise, ≈ 700) obtained from various values of α. They are randomly separated into training (70%) and test (30%) series. The network is a "Reservoir of computation (RC)" network coupled with a perceptron network (Fig. 3(b)). Computation Reservoir (RC) is fed

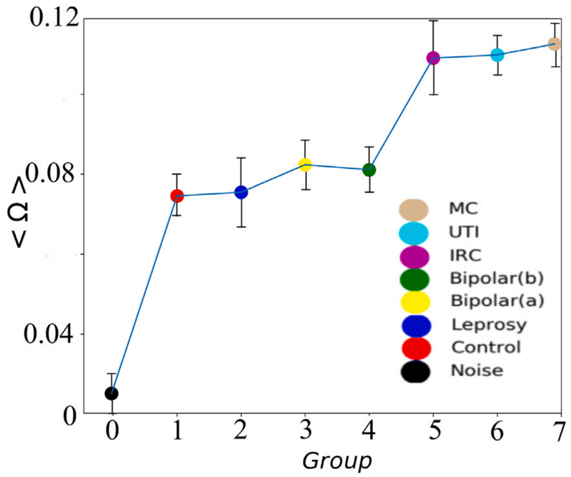

Fig. 8. Values of Ω as a function of each group. The higher value of Ω, the lower its variability. Control (volunteers), CRF (kidney diseases), (a) bipolar (patients before treatment), (b) bipolar (patients after treatment) and MC (brain death).

only with time series without the correlation quantifier (α). RC is used to learn the dynamics of each time series $\mathbf{x} = \{\mathbf{x}_t\}_{t=0}^{T}$ as follows: first project the time series with smaller dimension $\mathbf{x}(t)$ for a larger space, through the reservoir. Then a dimension reduction algorithm (PCA) projects the reservoir outlet into a smaller space. The RC output feeds a multilayer perceptron network (MLP) that correlates the network dynamics with the value of α, respectively. The network learning is done by regression, since the values of α are not integers. The error function metric is the mean-squared error,

$$\varepsilon = \frac{1}{N}\sum_{i=1}^{N}(\hat{\alpha}_i - \alpha_i)^2, \tag{4}$$

where $\hat{\alpha}$ is predicted by the network and α is the real value of the series, respectively. After training, we utilize the separate series for the test in the already trained network and compare the predicted values of α_i with the real values. The final error is given by $\varepsilon_{test} \approx 0.01$ and $\varepsilon_{training} \approx 0.001$. After the trained network, we apply the same network to the HRV signal to obtain the value of α of this same time series. With this value of α, we generate stochastic noise.

After the trained network, we apply the same network to the HRV signal to obtain the value of $\hat{\alpha}$ of this same time series. With this value of $\hat{\alpha}$, we generate stochastic noise of both the HRV signal and the noise with the same value of $\hat{\alpha}$.

From the entropies, we use a quantifier defined by [23],

$$\Omega(\hat{\alpha}) = \frac{|\,\overline{S}_{fn}(\hat{\alpha}) - \overline{S}\,|}{\overline{S}_{fn}(\hat{\alpha})}, \tag{5}$$

where $\overline{S}$ is the permutation entropy of the analyzed time series and $\overline{S}_{fn}(\hat{\alpha})$ is the permutation entropy of a flicker noise time series generated with the value of α returned by the recurrent neural network, $\hat{\alpha}$. The greater Ω, the lower the variability of the HRV signal.

In Fig. 8, we plot the mean value of Ω (difference between entropies) for each group of cardiac series. We observe that the stochastic noise has a small Ω value. The control group has a greater variability than people with brain death. The patients with bipolar disorders, after treatment, have an increase in their variability (decrease of Ω). The patients with chronic kidney disease and in the ICU are closer to patients with brain death. Analyzing the two approaches (t-SNE and entropy difference), both methodologies reach the same conclusion, showing that the variability of the HRV signal is a great indicator in the diagnosis of a patient. And all this analysis, has as its final application, the use of the algorithm AnSeCar [34].

4. Conclusions

In this work, we show that the measurement of the variability of the HRV signal is a great indicator of the patient's health status. The quantification of the variability can be directly related to the severe or less severe disease. This relationship is directly shown in the use of unsupervised learning, which groups the cardiac series according to their dynamic correlations. When we consider some stochastic noises, we observe that the patients with a more critical health status are farther away from these noises. The groups with a less severe health status show a strong relationship with the dynamics related to the stochastic noises. By means of the entropy difference of the patient groups, we verify that the variability of HRV signals is strongly related to the patient's status. The algorithm is able to capture the difference in the variability of HRV signals from the group of patients with bipolar disorder that received treatment. As a result of this work, we built an algorithm with medical application in the diagnosis of an individual's health status, measuring their HRV signal. This algorithm can be used as an entry point in the diagnosis of a patient. It is fast, does not need pre-treatment in the data, and has good accuracy. The next steps will be to obtain more HRV signals, so that we can improve the prediction of our algorithm.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The authors acknowledge the financial support from National Council for Scientific and Technological Development (CNPq), Brazil, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) and Fundação Araucária, Brazil.

Appendix A. Bidirectional deep read echo state (BDES) networks

Bidirectional architectures have been successfully applied in RNNs to extract time features from time series that also account for very distant dependencies in time, as it concatenates time series in both directions, recovering all information from the beginning of the time series, which would be lost if not bidirectional. The extraction of the dynamic characteristics of this time series can be used as input to another intelligent algorithm. This learning algorithm combines the training speed of Reservoir computing with the accuracy of trainable recurrent networks.

i. Reservoir computing

The reservoir computing (RC) has been used for modeling nonlinear time series. In the learning context, echo states are more common in RC models, where the input sequence is projected into a larger space through the use of the non-linear reservoir. Learning is accomplished through the application of simple linear techniques in the space of the reservoir.

The proposed architecture is adapted from [35], called bidirectional deep read ESN (BDESN), which combines the speed of the RC with the trainable precision of RNNs. The model is equipped with a bidirectional reservoir. Bidirectional architectures have been successfully applied in RNNs to extract temporal resources from the time series that have a very long time dependency.

ii. Reservoir

The reservoir acts as an encoder that generates the input representation in a larger space. The state produced by the reservoir brings all the dynamic information from the original input. This encoding is performed using the weights $\theta_{enc} = \{\mathbf{W^i}, \mathbf{W^h}\}$ and the dynamics of this process is performed by the following equation

$$\mathbf{h}(t) = (1-\alpha)\mathbf{x}(t) + \alpha f\left(\mathbf{W}^h\mathbf{h}(t-1) + \mathbf{W}^i\mathbf{x}(t) + \eta\right). \tag{6}$$

where $\mathbf{h}(t)$ is the time-dependent internal state, which combines the current input $\mathbf{x}(t)$ with the previous state $\mathbf{h}(t-1)$. The initial value of $\mathbf{h}(0)$ is zero. The function f is a non-linear activation function (tanh), $\mathbf{W}^h$ is the sparse matrix that defines the recurrent self-connects in the reservoir, and $\mathbf{W}^i$ defines the incoming connections. Both matrices are randomly generated and are not trained. The behavior of the reservoir is mainly controlled by five hyperparameters, that are: the size of the states N, the spectral radius ρ of W^h, the dimensioning of the inputs ω, the hyper leakage parameter α, and the noise η which is used for regularization in the reservoir. The η term represents additive white Gaussian noise with spherical covariance matrix and unit standard deviation. By means of an optimal fit of these hyperparameters, the reservoir produces rich dynamics and its internal states can be used to solve many prediction and classification tasks. The state generated by the reservoir $\mathbf{h}(t)$, after all inputs have been processed, is a high-dimensional representation that incorporates the temporal dependencies of $\mathbf{x}$. Since the reservoir exchanges its internal stability with a memory at time T [36], the state tends to lose information from the initial times. To get around this problem, we feed the same reservoir with the inverse order of the time series $\mathbf{x}' = \{\mathbf{x}_{T-t}\}_{t=0}^{T}$ and generate a new state $\mathbf{h}(t)'$ that is more influenced with the first inputs. The final resultant state is obtained by concatenating the two states, $\bar{\mathbf{h}}_T = [\mathbf{h}(t); \mathbf{h}(t)']$. The bidirectional reservoir has recently been used for time series prediction [37]. From the sequence of the RNN states generated over time,

$$\mathbf{H} = [\bar{h}(1), \ldots, \bar{\mathbf{h}}(T)], \tag{7}$$

it is possible to extract a representation $\mathbf{r}_X = r(\mathbf{H})$ of the input $\mathbf{x}$. The $\mathbf{r}_X$ vector brings us all the information about the characteristics of the $\mathbf{x}$ input, in this case the $\mathbf{r}_X$ vector is formed by the weights and bias learned when a later state is generated by the previous state, as shown in the equation

$$\mathbf{H}(t+1) = \mathbf{W}_r\mathbf{H}(t) + \mathbf{b}_r, \tag{8}$$

where $\mathbf{r}_X = \{\mathbf{W}_r, \mathbf{b}_r\} \in \mathbb{R}^{R(R+1)}$ and R is the number of neurons that form the reservoir. Learning is done through the Ridge method of the sklearn library.

In summary, all dynamic characteristics of the input are represented by the vector $\mathbf{r}_X$, which is formed by the weights $\{\mathbf{W}_r, \mathbf{b}_r\}$. After constructing $\mathbf{r}_X$, we can decode in the output space, which are the y classes for the classification case (Fig. 3(a)) or for the regression case (Fig. 3(b)). This decoding can be performed by

$$y = g(\mathbf{r}_X, \theta_{dec}), \tag{9}$$

where g is a multilayer perceptron network (Fig. 3(a)) and the θ_{dec} weights to be learned. This $\mathbf{r}_X$ state vector can also be used as input to an unsupervised learning algorithm (Fig. 3(c)).

As the reservoir has a large dimension due to the number of neurons, this takes an overfit and computational resources. The PCA [26] dimensionally reduces the states, showing a better performance. The PCA aims to reduce the feature space, choosing a space that better separates these features, facilitating the decision surface.

iii. Multilayer perceptron (MLP)

These state vectors with their reduced dimensions become the input vectors to the network (MLP), where the classification or regression takes place. The weights of these layers will undergo adjustments during training. At this point, a normal training of an MLP network is made.

These deep MLPs are known for their generalizability and adaptability, important characteristics for the problem at hand. Nowadays, deep layer networks can be efficiently trained using sophisticated regularization techniques and pre-training techniques that help to avoid overfit and null or explosive.

Appendix B. t-Stochastic Neighbor Embedding (t-SNE)

Unsupervised learning is a branch of Machine Learning that learns from test data which were not previously labeled, classified or categorized. Rather than responding to an operator's programming, unsupervised learning identifies similarities in data and reacts based on the presence or absence of such similarities in each new piece of data. One of the goals of this learning is to group in "clusters" the data that have strong correlation, allowing a visualization of the data without a "prior" knowledge of your characteristics. This learning serves as a starting point for the study of the dataset characteristics under analysis.

The t-Stochastic Neighbor Embedding (t-SNE) [38], also called non-parametric t-SNE, is a classical machine learning method for visualizing high-dimensional data. The idea of t-SNE is to map data points in the original high-dimensional space ($\mathscr{X} = \{x_1, x_2, \ldots, x_N\}$) to points in a low-dimensional space ($\mathscr{Y} = \{y_1, y_2, \ldots, y_N\}$), while keeping the similarity among the points. The map is determined by minimizing the KL-divergence between the similarity of data distributions in the high- and low-dimensional space.

In detail, the t-SNE defines the similarity between a high-dimensional data point x_i and another data point x_j by the following joint probability

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N} \tag{10}$$

where,

$$\begin{aligned} p_{j|i} &= \frac{e^{-\|x_i - x_j\|^2/2\sigma_i^2}}{\sum_{k\neq l} e^{-\|x_k - x_l\|^2/2\sigma_i^2}}, \\ p_{i|i} &= 0 \end{aligned} \tag{11}$$

and σ_i are parameters determined from the following quantity called perplexity of data x_i

$$Perp(P_i) = 2^{H(P_i)} \tag{12}$$

where $H(P_i) = \sum_j p_{j|i} \log_2 p_{j|i}$.

The value of σ_i is set so as to make $Perp(P_i)$ an user specified value (typically between 5 and 50). The similarity between the low-dimensional data points y_i and y_j is defined by the following equation using Student t-distribution with one degree of freedom [38]

$$\begin{aligned} q_{ji} &= \frac{(1+ \parallel y_i - y_j \parallel^2)^{-1}}{\sum_k \sum_{k\neq l}(1+ \parallel y_k - y_l \parallel^2)^{-1}}, \\ q_{i|i} &= 0 \end{aligned} \tag{13}$$

The t-SNE determines a low-dimensional point y_i corresponding to a data point x_i by iteratively minimizing the cost function $C(\{y_i\})$ defined as the Kullback–Leibler (KL) divergence between a joint probability distribution in the high- and low-dimensional data,

$$C(\{y_i\}) = \sum_i KL(P_i \parallel Q_i) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}, \tag{14}$$

where P_i represents the probability distribution over all points given by x_i and Q_i represents the probability distribution over all points given byy_i.

Table 2
Hyper parameters of the reservoir.

Accuracy	α	ρ	ω	η
0.62	0.7500	0.9700	0.90	0.0011
0.74	1.0000	0.9690	0.92	0.0020
0.52	0.8000	0.8123	1.00	0.0021
0.79	0.7200	0.9900	0.94	0.0012
0.80	0.7500	0.7200	0.98	0.0014
0.42	0.6000	0.9982	0.90	0.0200

The gradient of the cost function with respect to the set of variables y_i is given by

$$\frac{\partial C}{\partial y_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \parallel y_k - y_l \parallel^2)^{-1}, \tag{15}$$

where $p \approx q$, we have $\frac{\partial C}{\partial y_i} \approx 0$.

The gradient is initialized by randomly sampling the mapped points. In order to speed up the optimization and avoid poor local minima on the error surface, a relatively large momentum term is added to the gradient,

$$\mathscr{Y}^{(t)} = \mathscr{Y}^{(t-1)} + \eta\frac{\partial C}{\partial y_i} + a(t)(\mathscr{Y}(t-1) + \mathscr{Y}(t-2)), \tag{16}$$

where $\mathscr{Y}^{(t)}$ indicates iteration solution t, η is the learning rate, and $a(t)$ represents the moment in t.

Appendix C. Sensitivity to hyperparameters

As is known [39,40], the dependence of the prediction results on the hyperparameters can be quite sensitive. We have used the Bayesian optimization method [41,42] that is contained in the PYTHON package "skopt"[43]. An issue is that the optimization algorithm typically gives multiple sets of hyperparameter values. We consider these hyperparameter values to train multiple reservoirs and obtain the average validation RMSE that can be fed back to the Bayesian algorithm. For each set of the hyperparameter values, we repeat the training and validation processes multiple times with different random realizations of the reservoir to reduce the fluctuations in RMSE. After several hundreds of iterations of the Bayesian algorithm, we choose the hyperparameter values with the lowest validation RMSE in all the iterations (not necessarily the hyperparameter values from the last iteration).

A deficiency of the Bayesian optimization algorithm is that sometimes it generates solutions that are not optimal. It occurs when the solution trajectory is trapped in a local minimum of the landscape of the cost function, especially when the RMSE from the validation process has large fluctuations. An empirical solution is to run the whole Bayesian optimization process independently a number of times and choose the best result with the smallest error. Table 2 shows some accuracy values that depend on the hyperparameters.

References

[1] Dampney RAL. Functional organization of central pathways regulating the carviovascular system. Physiol Rev 1994;74:323–64.

[2] Fatisson J, Oswald V, Lalonde F. Influence diagram of physiological and environmental factors affecting heart rate variability: an extended literature overview. Heart Int 2016;11:e32–40.

[3] Acharya UR, Joseph KP, Kannathal N, Lim CM, Suri JS. Heart rate variability: a review. Med Biol Eng Comput 2006;44:1031–51.

[4] Chua KC, Chandran V, Acharya UR, Lim CM. Computer-based analysis of cardiac state using entropies, recurrence plots and Poincare geometry. J Med Eng Technol 2008;32:263–72.

[5] Malik M. Heart variability. Curr Opin Cardiol 1998;13:36–44.

[6] Vanderlei LCM, Pastre CM, Hoshi RA, de Carvalho TD, de Godoy MF. Basic notions of heart rate variability and its clinical applicability. Rev Bras Cir Cardiovasc 2009;24:205–17.

[7] Meyerfeldt U, Wessel N, Schütt H, Selbig D, Schumann A, Voss A, et al. Heart rate variability before the onset of ventricular tachycardia: differences between slow and fast arrhythmias. Int J Cardiol 2002;84:141–51.

[8] Marwan N, Wessel N, Meyerfeldt U, Schirdewan A, Kurths J. Recurrence-plot-based measures of complexity and their application to heart-rate-variability data. Phys Rev E 2002;66:026702.

[9] Moridani MK, Setarehdan SK, Nasrabadi AM, Hajinasrollah E. Analysis of heart rate variability as a predictor of mortality in cardiovascular patients of intensive care unit. Biocybern Biomed Eng 2015;35:217–26.

[10] Liu YLA, Macau EEN, Barroso JJ, Silva JDS, Guimarães-Filho ZOA, Caldas IL, et al. Tendências em matemática aplicada e computacional, vol. 9. 2008, p. 255–64, Número 2.

[11] Samuel AL. Some studies in machine learning using the game of checkers. IBM J Res Dev 1959;44:206–26.

[12] Jordan MI, Mitchell TM. Machine learnings: Trends, perspectives, and prospects. Science 2015;349:255–60.

[13] Schimidt J, Marques MRG, Botti S, Marques MAL. Recent advances and applications of machine learning in solid-state materials science. Npj Comput Mater 2019;5:83.

[14] Jain P, Coogan SCP, Subramanian SG, Crowley M, Taylor S, Flannigan MD. A review of machine learning applications in wildfire science and management. Environ Rev 2020;28:478–505.

[15] Libbrecht MW, Noble WS. Machine learning applications in genetics and genomics. Nat Rev Genet 2015;16:321–32.

[16] Sakai K, Yamada K. Machine learning studies on major brain diseases: 5-year trends of 2014–2018. Jpn J Radiol 2019;37:34–72.

[17] Baby PS, Vital TP. Statistical analysis and predicting kidney diseases using machine learning algorithms. Int J Eng Res Technol 2015;4:206–10.

[18] Herry CL, Soares HMF, Schuler-Faccini L, Frasch MG. Machine learning model on heart rate variability metrics identifies asymptomatic toddlers exposed to zika virus during pregnancy. Physiol Meas 2021;42:055008.

[19] Krittanawong C, Virk HUH, Bangalore S, Wang Z, Johnson KW, Pinotti R, et al. Machine learning prediction in cardiovascular diseases: a meta-analysis. Sci Rep 2020;10:16057.

[20] Parthiban G, Srivatsa SK. Applying machine learning methods in diagnosing heart disease for diabetic patients. Int J Appl Inf Syst 2012;3:25–30.

[21] Ali MM, Paul BK, Ahmed K, Bui FM, Quinn JMW, Moni MA. Heart disease prediction using supervised machine learning algorithms: Performance analysis and comparison. Comput Biol Med 2021;136:104672.

[22] Dua S, Du X, Sree SV, T. Ahamed VI. Novel classification on coronary artery disease using heart rate variability analysis. J Mech Med Biol 2012;12:1240017.

[23] Boaretto BRR, Budzinski RC, Rossi KL. Discriminating chaotic and stochastic time series using permutation entropy and artificial neural networks. Sci Rep 2021;11:15789.

[24] Lukosevicius M, Jaeger H. Reservoir computing approaches to recurrent neural network training. Comp Sci Rev 2009;3:127–49.

[25] Bianchi FM, Scardapane S, Lokse S, Jenssen R. Reservoir computing approaches for representation and classification of multivariate time series. IEEE Trans Neural Netw Learn Syst 2021;32:2169–79.

[26] Tharwat A. Principal component analysis-a tutorial. Intern J Appl Pattern Recognit 2016;3:197–240.

[27] Bengio Y, Lamblin P, Popovici D, Larochelle H. Greedy layer-wise training of deep networks. Adv Neural Inf Process Syst 2007;153–60.

[28] Bandt C, Pompe B. Permutation entropy: A natural complexity measure for time series. Phys Rev Lett 2002;88:174102.

[29] Boaretto BR, Budzinski RC, Rossi KL, Prado TL, Lopes SR, Masoller C. Evaluating temporal correlations in time series using permutation entropy, ordinal probabilities and machine learning. Entropy 2021;23:1025.

[30] Zunino L, Soriano MC, Fischer I, Rosso OA, Mirasso CR. Classifying cardiac biosignals using ordinal pattern statistics and symbolic dynamics. Comput Biol Med 2012;42:319–27.

[31] Parlitz U, et al. Classifying cardiac biosignals using ordinal pattern statistics and symbolic dynamics. Comput Biol Med 2012;42:319–27.

[32] Liu R, Wu T, Mozafari B. Adam with bandit sampling for deep learning. Adv Neural Inf Process Syst 2020;33:5393–404.

[33] da Silva ST, Ansecar. GitHub repository. 2021, https://github.com/10618610/Ansecar.

[34] da Silva ST, Ansecar. Webapp. 2021, https://share.streamlit.io/10618610/webapp/main/AnSeCar.py.

[35] Bianchi FM, Scardapane S, Lokse S, Jenssen R. Bidirectional deep-readout echo state networks. In: European symposium on artificial neural networks, computational intelligence and machine learning. 2018.

[36] Bianchi FM, Livi L, Alippi C. Investigating echo state networks dynamics by means of recurrence analysis. IEEE Trans Neural Netw Learn Syst 2018;29:427–39.

[37] Rodan A, Sheta AF, Faris H. Bidirectional reservoir networks trained using SVM+ privileged information for manufacturing process modeling. Soft Comput 2017;21:6811–24.

[38] der Maaten LV, Hinton G. Visualizing data using t-SNE. J Mach Learn Res 2008;2579–605.

[39] Haynes ND, Soriano MC, Rosin DP, Fischer I, Gauthier DJ. Reservoir computing with a single time-delay autonomous boolean node. Phys Rev E 2015;91. 020801(R).

[40] Fan H, Jiang J, Zhang C, Wang X, Lai Y-C. Long-term prediction of chaotic systems with machine learning. Phys Rev Res 2020;2. 012080(R).

[41] Griffith A, Pomerance A, Gauthier DJ. Forecasting chaotic systems with very low connectivity reservoir computers. Chaos 2019;29:123108.

[42] Kong Ling-Wei, Fan Hua-Wei, Grebogi Celso, Lai Ying-Cheng. Phys Rev Res 2021;3:013090.

[43] https://github.com/scikit-optimize/scikit-optimize.