Combining Algorithmic Information Dynamics Concepts and Machine Learning for Electroencephalography Analysis: What Can We Get?

Victor Iapascurta
Department of Anesthesia and Intensive Care
N. Testemitanu State University of Medicine and Pharmacy
165, Stefan cel Mare si Sfant, Bd., MD-2004
Chisinau, Republic of Moldova
viapascurta@yahoo.com

Electroencephalography (EEG) as an example of electrophysiological monitoring methods has a rather long history of successful application for the diagnosis and treatment of diseases, and this success would not have been possible without effective methods of mathematical, and more recently, computer analysis. Most of these methods are based on statistics. Among the methods of EEG analysis, there is a group of methods that use different versions of Shannon's entropy estimation as a "main component" and that do not differ significantly from traditional statistical approaches. Despite the external similarity, another approach is to use the Kolmogorov–Chaitin definition of complexity and the concepts of algorithmic information dynamics. The algorithmic dynamics toolbox includes techniques (e.g., block decomposition method) that appear to be applicable to EEG analysis. The current paper is an attempt to use the block decomposition method along with the recent addition to the management of EEG data provided by machine learning, with the ultimate goal of making this data more useful to researchers and medical practitioners.

Keywords: biomedical signal processing; electroencephalography; algorithmic complexity; algorithmic information dynamics; block decomposition method; machine learning; deep learning

1. Introduction

Electroencephalography (EEG) as a method of monitoring electrical activity of the brain has been successfully used for over one hundred years [1]. In a clinical setting, EEG refers to the recording of the brain's electrical activity over a period of time, using multiple electrodes placed on the scalp or implanted in the brain tissue, the latter known as intracranial EEG (iEEG) [1, 2].

The goals of using EEG as a monitoring method can be summarized as: (*a*) to help researchers gain a better understanding of the brain; (*b*) to assist physicians in diagnosis and treatment choices; and (*c*) to boost brain-computer interface (BCI) technology [3].

There are many ways to roughly categorize EEG analysis methods. As shown in a survey article [2], most EEG analysis methods fall into four categories: (*i*) time domain; (*ii*) frequency domain; (*iii*) time-frequency domain; and (*iv*) nonlinear methods. There are also later methods, including machine learning (ML). As for specific mathematical signal analysis methods, there is a multitude of approaches in all of the domains listed above: linear prediction (LP) and independent component analysis (ICA), fast Fourier transform (FFT), autoregressive (AR) methods, short-time Fourier transform (STFT), wavelet transform (WT) and others.

Since the EEG signal is far from stationarity and may contain much noise, the linear methods of analysis were thought not to be the best choice, at least in some situations. Nonlinear dynamical analysis has been a powerful approach for understanding these physiological signals. It was observed that nonlinear dynamics theory would be a better approach than traditional time domain and frequency domain methods for analyzing and characterizing the EEG signals. The collection of nonlinear methods also looks impressive: higher-order spectra (HOS) techniques, phase space plot (PPS) methods, correlation dimension (CD) and fractional dimension (FD) methods, largest Lyapunov exponent (LLE), entropy estimators and others.

Among the nonlinear methods, there is a group of entropy estimators (e.g., spectral entropy (SEn), approximate entropy (ApEn), sample entropy (SampEn)). Most of them are based on Shannon's entropy, which is also presented as a measure of algorithmic complexity (AC) [2, 4].

Recent researchers, however, question the use of Shannon's entropy as the best (or sometimes even appropriate) estimation for AC, and the Kolmogorov–Chaitin definition of AC is used instead [5–7].

The current paper is trying to combine the algorithmic complexity (by the Kolmogorov–Chaitin approach) as a metric and data representation method before data is fed to an ML algorithm.

This research is based on data generated by the epileptic brain as well as a separate dataset from healthy individuals with a similar final goal: discriminating between interictal and preictal EEG clips (in the case of epileptic subjects) or classifying healthy subjects depending on their abilities concerning certain mental activities (e.g., arithmetic counting).

2. Materials and Methods

2.1 Data

There are two sets of data used in this study: the epileptic set [8]; and the non-epileptic/healthy set [9]. The data in the epileptic set is provided by the National Institutes of Health, the Epilepsy Foundation and the American Epilepsy Society for the international competition "American Epilepsy Society Seizure Prediction Challenge," which does not prohibit the use of the data for education. Its goal was to identify the best model for discriminating preictal versus interictal (between seizures) iEEG clips. The competition was hosted on the Kaggle platform [8]. Intracranial EEG (iEEG) was recorded from five dogs with naturally occurring epilepsy and two humans undergoing iEEG monitoring. For canine subjects, the iEEG signal was sampled from 16 electrodes at 400 Hz and voltages were referenced to the group average. These recordings span multiple months up to a year and contain up to 100 seizures in some dogs. The sample rate for human patients was 5000 Hz and the recorded voltages were referenced to an electrode outside the skull; the monitoring period was for up to a week. For canine subjects, 16 channels were recorded, except dog-5, who had 15 channels. For humans, patient-1 and patient-2 had 15 and 24 channels, respectively. The data is organized into 10-minute-long clips of preictal and interictal activity. Training data is grouped into one-hour sequences; the timing of the test clips is unknown. The preictal period starts one hour prior to seizure onset with a five-minute horizon: from 1:05 to 0:05 before seizure.

This dataset consists of a total of 8012 subsets, each one representing a 10-minute multichannel EEG recording clip. Out of this, 4077 are included in the training set (3766 as interictal and 311 as preictal segments) and 3935 sets represent the test set. The human portion consists of 128 and 345 subsets (as training and test, respectively).

As described in the coming sections, the current research used data from dog-1 and dog-2 and one of the humans. The ML system was built using the dog-2 set, since this training set is one of the largest and most balanced sets (i.e., interictal (500) samples versus preictal (42) samples).

The non-epileptic/healthy set contains EEG recordings of 36 healthy volunteers before and during the performance of mental arithmetic tasks. The arithmetic task was the serial subtraction of two numbers. Each trial started with the oral communication of 4-digit (minuend) and 2-digit (subtrahend) numbers (e.g., 3141 and 42). More details concerning this set are presented in Section 3, where data processing steps are described.

2.2 Data Processing Techniques

A central role in data processing flow is attributed to the estimation of algorithmic (Kolmogorov–Chaitin) complexity performed via the block decomposition method, coming from the field of algorithmic information dynamics [5, 10].

Of primary importance here is the definition of algorithmic (Kolmogorov–Chaitin or program-size) complexity (Kolmogorov, 1965; Chaitin, 1969) [10]:

$$K_T(s) = \min\{|p|, T(p) = s\}, \tag{1}$$

that is, the length of the shortest program p that outputs the string s running on a universal Turing machine T.

Algorithmic information dynamics (AID) is an emerging field of complexity science based on algorithmic information theory, which comprises the literature based on the concept of Kolmogorov–Chaitin complexity and related concepts such as algorithmic probability, compression, optimal inference, the universal distribution, Levin's semi-measure and others.

AID strives to search for solutions to fundamental questions about causality: why a particular set of circumstances leads to a particular outcome. In this aspect it essentially differs from traditional statistics. As an applied science, AID is a new type of discrete calculus based on computer programming and aimed at studying causation by generating mechanistic models to help find first principles of physical phenomena, building up the next generation of machine learning [11].

In the AID toolkit, there is a special tool for providing reliable estimations to uncomputable functions, namely the online algorithmic complexity calculator (OACC) [12], which provides estimations of algorithmic complexity and algorithmic probability for short and long strings and for two-dimensional arrays better than any other traditional tool, none of which can capture any algorithmic content beyond simple statistical patterns. The OACC uses the BDM method [12], which is based upon algorithmic probability defined by the coding theorem method (CTM) [13, 14]:

$$\text{BDM} = \sum_{n=1}^{n} \text{CTM}(\text{block}_i) + \log_2(|\text{block}_i|) \tag{2}$$

The OACC is available as an online version [15] as well as standalone packages in R and a number of other languages, including Wolfram Language [12], and it is used for calculations throughout this paper.

Every file (subset) in the original data is a .mat file describing the EEG signal voltage variations for n channels for a 10-minute duration. The subset is unfolded and a matrix with columns representing n channels/electrodes and rows denoting observations of EEG signal variation over time corresponding to particular channels is generated. The resulting matrix is split into a number of 16×16 (or 15×15, depending on the number of channels/electrodes) matrices, keeping the tie with the electrodes and time. These small matrices are binarized (using the BASCA method, Binarize package in R). The BDM value for every such matrix is calculated, with BDM values arranged over the time axis, obtaining time series that describe BDM value variation over time. Since the volume of data BDM value is to be calculated on is large, neither online nor the regular standalone version of the OACC is suitable. For the purpose of this paper, the "core" of the R version of OACC was extracted and integrated into the data processing flow. The binarization and BDM calculation on this large dataset are computationally expensive. To address this, the EC 2 Amazon Web Service is being used. Taking into account the sampling rate of 400 Hz and the dimension of a small matrix (e.g., 16×16) described earlier, with every 10-minute subset there are 15 000 matrices generated (i.e., 400Hz * 600 seconds/16). Once the BDM value is calculated, it can be plotted along the time axis with a total number of steps equal to 15 000, which represents the BDM value dynamics over time for a particular patient. A detailed explanation of the data processing steps is present in [16].

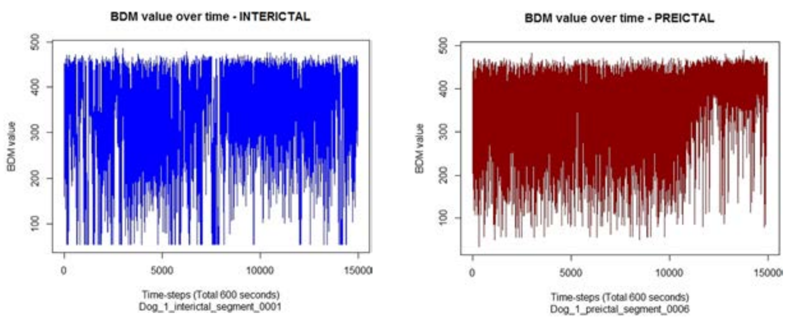
3. Results

3.1 Preliminaries

The data representing the original EEG signal from multiple electrodes is split into square matrices, keeping the ties with electrodes and time, and a matrix BDM is calculated after binarization. The data obtained this way resembles a time-series–like structure (or univariate time series) with the x axis for time and the y axis for BDM value (e.g., Figure 1).

A simple visualization can provide a general view concerning the difference in BDM value trace of EEG segments coming from distinct classes ("interictal" versus "preictal"). This is valid only for typical cases, while most segments cannot be classified based on simple visualization. In this way, conventional exploratory data analysis does not provide too much information.

Many natural processes are of a nonstationary nature. A common way of representing them is through time series [17], which will also be nonstationary in this case. A stationary time series is one whose statistical properties such as mean, variance, autocorrelation and others are all constant over time. The processes in the brain represented by the EEG signal are considered to be far from stationary, and accordingly, the EEG signals are treated as nonstationary time series.



(a) (b)

Figure 1. Time-series–like BDM value over time: (a) data from an interictal clip; and (b) data from a preictal clip.

Time series consisting of BDM value over time (hereafter denoted as BDM-TS), calculated based on EEG, should be expected to inherit the nonstationarity, which can be checked by a number of methods.

As can be seen in Table 1, there are ambiguities between the ADF and KPSS tests. Without going into detail, we can assume that the structure of the BDM-TS apparently has a rather complex nature, which can be difficult to capture using traditional analytical tools.

Test	Test *P*-Value	Threshold *P*-Value
ADF	<=0.01	0.05
KPSS	from <=0.01 to 0.1	0.05

Table 1. The results of applying augmented Dickey–Fuller (ADF) and Kwiatkowski–Phillips–Schmidt-Shin (KPSS) tests to 1000 BDM-TS based on the dog-2 test set.

Most statistical forecasting methods are based on the assumption that the time series can be rendered approximately stationary (i.e., stationarized) through the use of mathematical transformations. But one of the shortcomings of this approach is that tools using the "stationarizing" and similar approaches can "erase"/skip any nonlinear and algorithmic regularity if it does not show a statistical property. This may be especially valid when applying such methods to BDM-TS by making the most important characteristics of BDM as a metric of nonlinearity and algorithmic complexity vanish.

For solving the task of classifying the BDM-TS, a number of tools designed for time series analysis (e.g., dynamic time wrapping (DTW), k-means clustering and anomaly detection) were applied. As one of the most promising, the text analysis (TA) approach was selected. It includes some tools and procedures/algorithms that are specially adapted for the purposes of this research. These methods were used as separate exploratory tools and, finally, as ways to generate features to be fed to neural network (NN) models.

The current research is limited to using NN models [17] for ML and up to a certain depth, three main types of neural networks (NNs) were used: namely, multilayer perceptron (MLP) or fully connected version, convolutional neural networks (CNNs) and recurrent neural networks (RNNs), the last mainly as the long short-term memory (LSTM) version. MLP is built on the H2O.ai platform, and for CNN and RNN the MXNet R package is used.

Initial feeding of "raw" BDM-TS to different types of networks for a binomial classification did not provide satisfactory results: the best accuracy (slightly over 0.6) was with MLP/H2O, while RNN-LSTM refused to converge. Although it provided some useful insights/intuitions for further work (which is described below), this "raw-data approach" was not further developed.

3.2 Initial Exploratory Data Analysis

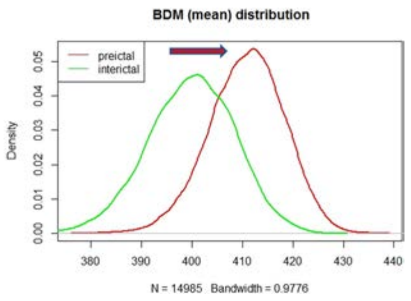
Since exploratory data analysis (EDA) is critical to ML, especially in terms of preparing the data for ML, EDA can help identify the preparatory steps.

After the simple visualization of BDM-TS described in Section 3.1, an approach to continue with was density distribution by computing BDM density estimates for BDM-TS using a Gaussian kernel (base R density() function). The datasets used for EDA represent the BDM-TS calculated on EEG segments of 10 minutes duration that are present in the training sets (dog-1, dog-2 and patient-1) from the Kaggle competition [8]. These sets consist of preictal and interictal subsets. For each subset and in some cases for individual segments, simple statistics were calculated.

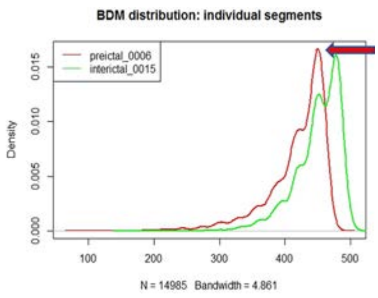
Figure 2 shows the density distribution for the dog-2 training set. BDM was averaged for every time step. On the left plot, there can be noticed an evident shift to the right of the preictal curve. The sets used for estimations and presented on the plot consist of 42 preictal subsets (or full preictal set in the case of dog-2 subject) and 42 interictal subsets randomly sampled out of 500 interictal segments (or from the dog-2 full interictal set).

But visualizing the same metric for a pair of individual segments from the dog-2 training set (i.e., preictal segment 0006 and interictal segment 0015) reveals an opposite direction shift (see Figure 2(b)). Thus, the simple mean BDM value seems to hardly bear any discriminative power. On the other hand, this does not exclude the existence of nonlinear relations with discriminating properties in the data.

In order to identify other features that can be of help for discriminating purpose, more complex approaches have been tried.



(a)



(b)

Figure 2. BDM distribution for: (a) the entire dog-2 training set; and (b) separate segments/clips. N denotes the number of observations. Bandwidth is selected automatically with the "nrd0" method.

3.3 Encoding BDM Values over Time as Images

This approach is intended for two purposes: (*a*) to get some useful details concerning "inner" data structure that may be of help for the discriminatory task; and (*b*) to use these images as an appropriate form of data representation to be fed to a CNN. Two methods were tried: recurrence plots and Gramian angular fields.

A recurrence plot (RP) [18, 19] is a technique of nonlinear data analysis and represents a visualization of a square matrix, in which the matrix elements correspond to those times at which the state of a dynamical system recurs. The RP reveals all the times when the phase space trajectory of the system visits roughly the same area in the phase space. Figure 3 presents some images obtained by building RP based on BDM-TS.

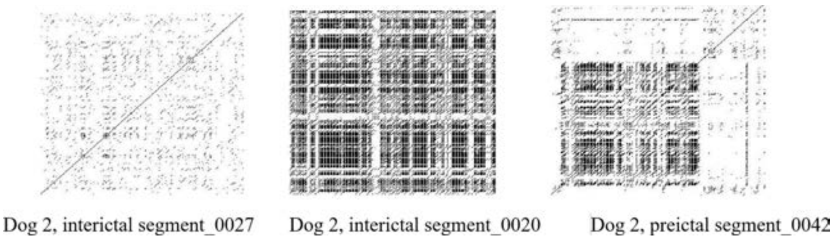


Figure 3. The recurrence plots (RPs) of three BDM-TSs generated based on three EEG clips from dog-2 set.

Gramian angular fields (GAF) [20, 21] is another method of encoding time series as images with the aim to apply convolutional neural networks (Figure 4). Using GAF, a time series can be represented in a polar coordinate system instead of the typical Cartesian coordinates.

In the Gramian matrix on which GAF is based, each element is actually the cosine of the summation of angles. The same BDM-TSs used earlier are transformed using GAF. As can be noticed, there are some similarities between the corresponding images, the GAF method containing an additional "color" dimension.

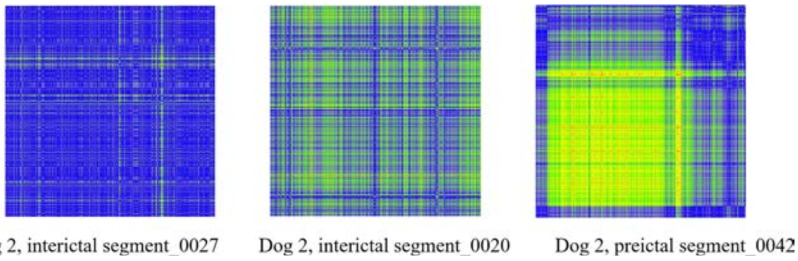


Figure 4. The Gramian angular fields for three BDM-TSs generated based on three EEG clips from dog-2 set.

Both methods indicate the presence of highly diverse structures in the data based on the texture of the images: from poorly correlated stochastic data (single dots on RP left image above) to changing states with transitions resembling laminar/recurrent states (vertical and horizontal lines on RP image in the middle) and abrupt changes (the image on the right).

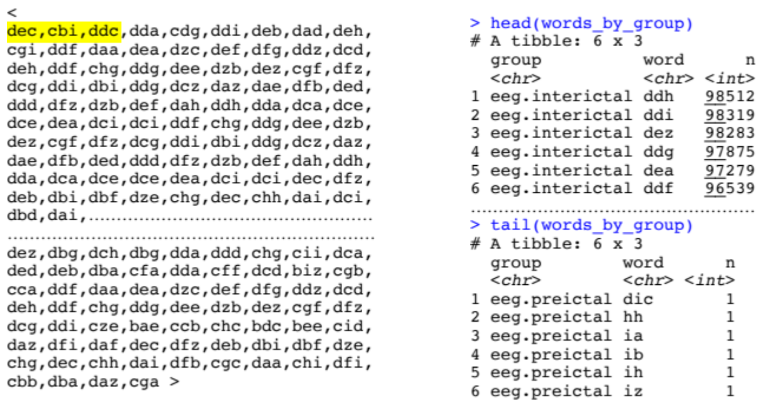
3.4 Text Analysis Approach

This section describes the use of some text analysis tools that proved to be quite effective in solving the problem of finding the possibilities of using BDM-TS in the workflow to achieve the specific goal of classifying EEG segments into two groups: preictal versus interictal.

The first step in applying tidy text analysis to BDM-TS consists of preparing the data to be accepted by the respective tools (e.g., R packages tidytext, tdyr, stringr and topicmodels). This includes converting the BDM-TS to a specific text format. For this, the BDM values in every TS were converted to text by a specially designed but simple function that converts numbers to letters. Since the BDM values are doubles with a number of decimals, prior to conversion the BDM values were rounded to units. After conversion, a flat text file containing a collection of words corresponding to rounded BDM values in the original file was generated, as shown in Figure 5(a).

The text organized in this way is tokenized, cleaned and grouped (into interictal and preictal), with the resulting data frame in the format shown in Figure 5(b).

Data is split into interictal and preictal subsets and words/letters are converted back to numbers, and as the next step, the corresponding density plots are generated (Figures 6 and 7).

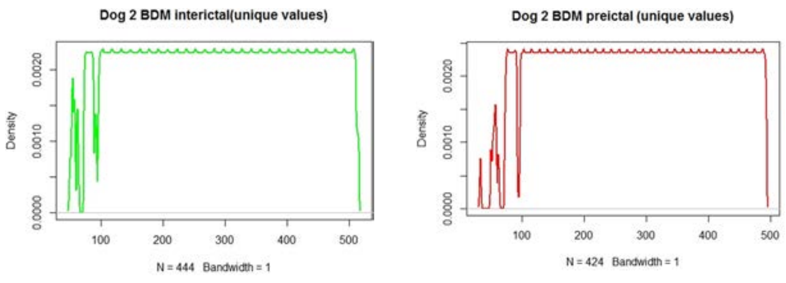


```
<
dec,cbi,ddc,dda,cdg,ddi,deb,dad,deh,
cgi,ddf,daa,dea,dzc,def,dfg,ddz,dcd,
deh,ddf,chg,ddg,dee,dzb,dez,cgf,dfz,
dcg,ddi,dbi,ddg,dcz,daz,dae,dfb,ded,
ddd,dfz,dzb,def,dah,ddh,dda,dca,dce,
dce,dea,dci,dci,ddf,chg,ddg,dee,dzb,
dez,cgf,dfz,dcg,ddi,dbi,ddg,dcz,daz,
dae,dfb,ded,ddd,dfz,dzb,def,dah,ddh,
dda,dca,dce,dce,dea,dci,dci,dec,dfz,
deb,dbi,dbf,dze,chg,dec,chh,dai,dci,
dbd,dai,..........................................
.....................................................
dez,dbg,dch,dbg,dda,ddd,chg,cii,dca,
ded,deb,dba,cfa,dda,cff,dcd,biz,cgb,
cca,ddf,daa,dea,dzc,def,dfg,ddz,dcd,
deh,ddf,chg,ddg,dee,dzb,dez,cgf,dfz,
dcg,ddi,cze,bae,ccb,chc,bdc,bee,cid,
daz,dfi,daf,dec,dfz,deb,dbi,dbf,dze,
chg,dec,chh,dai,dfb,cgc,daa,chi,dfi,
cbb,dba,daz,cga >
```

```
> head(words_by_group)
# A tibble: 6 x 3
  group           word      n
  <chr>           <chr> <int>
1 eeg.interictal ddh    98512
2 eeg.interictal ddi    98319
3 eeg.interictal dez    98283
4 eeg.interictal ddg    97875
5 eeg.interictal dea    97279
6 eeg.interictal ddf    96539
.......................................
> tail(words_by_group)
# A tibble: 6 x 3
  group         word      n
  <chr>         <chr> <int>
1 eeg.preictal dic        1
2 eeg.preictal hh         1
3 eeg.preictal ia         1
4 eeg.preictal ib         1
5 eeg.preictal ih         1
6 eeg.preictal iz         1
```

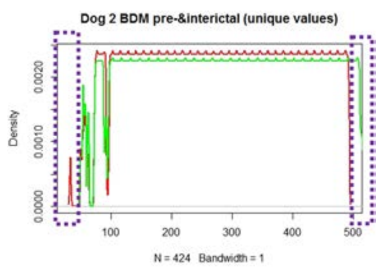
(a) (b)

Figure 5. (a) A fragment of the original BDM-TS file converted to text in case of dog-2 preictal segment 0006: dec encodes a BDM value of 453, cbi encodes 329, ddc encodes 443 and so forth. Total word count equals 14985 (i.e., the number of observations in this time series). (b) The words grouped by class (i.e., eeg.intericatal/eeg.preictal). The head and tail of the tibble data frame are shown. The "word" column contains unique BDM values converted to words and the "n" column shows the frequency of occurrence of the corresponding word in the dataset.

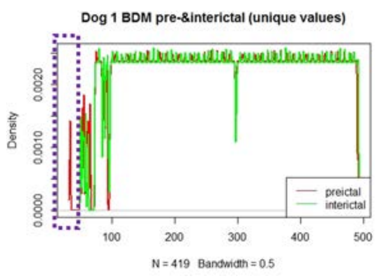


(a) (b)

Figure 6. Plots representing dog-2 BDM unique values (after conversion of words to numbers). These plots show: (a) the unique values occurrence for the interictal; and (b) preictal sets in dog-2.



(a)

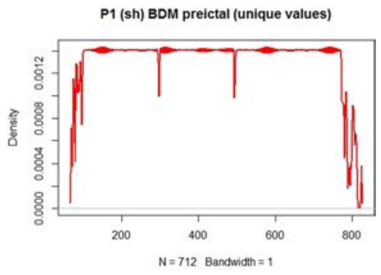


(b)

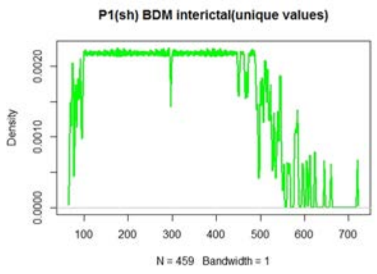
Figure 7. Plots representing dog-2 and dog-1 BDM unique values (after conversion of words to numbers). Superposition of the previous plots (i.e., in Figure 6) with the parts that differ (and can be used to infer the difference that would count for discriminative purpose), marked by the purple dotted frames: (a) dog-2; (b) dog-1.

Although for patient-1 the set of interictal and preictal segments is much smaller than the sets in the cases of dog-1 and especially dog-2, this is to some extent compensated by higher EEG resolution (5000 Hz versus 400 Hz) providing similar final data volumes. On the original EEG files of patient-1, the BDM value was estimated using 3×3 (15×15) matrices conditioned by the number of electrodes (15 channels versus 16 with the dogs). The resulting BDM-TSs were aggregated by summing every two cells/time step values. The plots for these final BDM-TSs are slightly different from the ones for the dogs (Figures 8 and 9).

As can be noticed on the plots, the BDM value interval that can serve for a discriminating purpose is the one starting with BDM approximately 720 and higher, denoted by the purple frame.

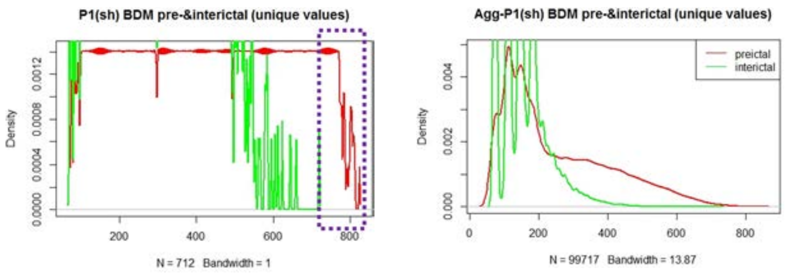


(a)



(b)

Figure 8. Plots representing patient-1 dataset BDM unique values (after conversion of words to numbers): (a) the preictal subset; and (b) the interictal subset.



(a) (b)

Figure 9. Plots representing patient-1 dataset BDM unique values (after conversion of words to numbers. (a) The superposition of previous plots (i.e., in Figure 8), with the purple frame denoting the difference that can potentially have discriminative value. (b) The BDM density distribution for the full set of BDM values in this set by replicating these values according to their frequency in the set.

In fact, some of the plotting could be performed based on numeric data, not necessarily converting original BDM-TS data to a letter/text format and then back to numeric. But text mining tools (that require these number-to-letter transformations) can provide some additional benefits that can be of help in solving the discriminating/classification task. In addition, this format can be used for building a DL model, which is described in Section 3.5).

Among TA tools, there is latent Dirichlet allocation (LDA), which is based on Dirichlet distribution and works as follows:

It treats every collection of words (i.e., documents) as a mixture of topics and every topic as a mixture of words. This allows documents to “overlap” each other in terms of content, rather than being separated into discrete groups, in a way that mirrors typical use of natural language. LDA is a mathematical method for estimating both of these (i.e., mixture of topics and mixture of words) at the same time: finding the mixture of words that is associated with each topic, while also determining the mixture of topics that describes each document. In our case, original BDM-TS files converted to text files serve as documents and BDM values in them, converted to letters, as words. Using the R package topicmodels, a model with two topics is generated. During this procedure, the per-topic-per-word probabilities, called β, can be extracted from the model. Figure 10 shows the top 30 terms, which represent the unique BDM values included in each of these topics, and we can see the most common words (which are the most common BDM values) for each topic.

After converting these words back to numbers and plotting the distribution of the top 100 BDM values for each topic, we get the plots in Figures 11 and 12.

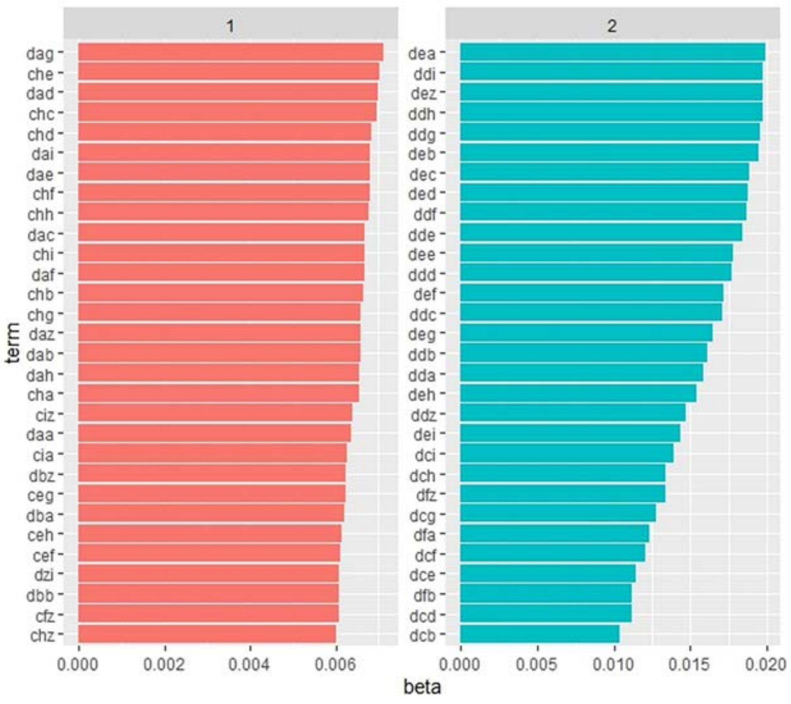


Figure 10. Top 30 terms (BDM unique values). These are terms (i.e., BDM values as "words") that are most common within each topic.

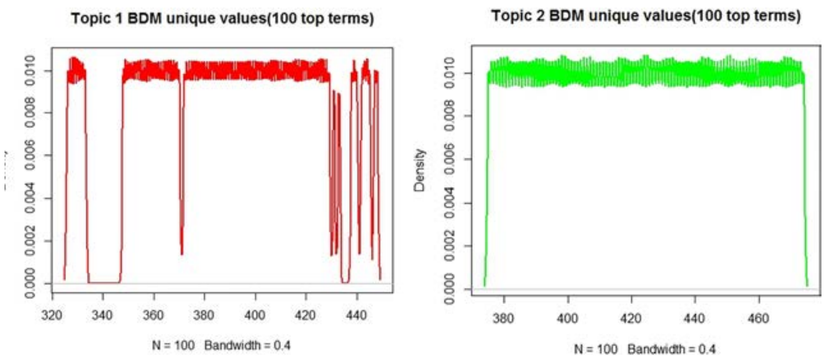


Figure 11. Topic 1 and topic 2 BDM unique values density distribution.

With LDA modeling, words (in our case BDM values as digits converted to letters) are the observable variables, and the other variables are latent variables. A sparse Dirichlet prior is used to model the topic-word distribution, following the intuition that the probability distribution over words in a topic is skewed, so that only a small set of words has high probability.

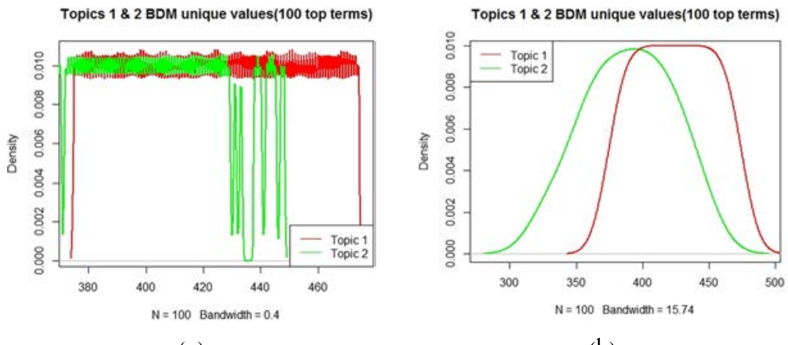


Figure 12. (a) The superposition of the plots in Figure 11. Non-overlapping parts of the plots represent the difference between these plots. (b) The same superposition with a larger bandwidth (15.74 versus 0.4), which makes the skewedness more obvious.

The difference between interictal and preictal segments using topics can be visualized using box plots (Figure 13), and although the extreme BDM values are not accounted for, we can see that for the preictal set there is a unique situation when the highest priority words of the topics are located far away from each other (red frames), while with the interictal set, an overlap is present (green frame).

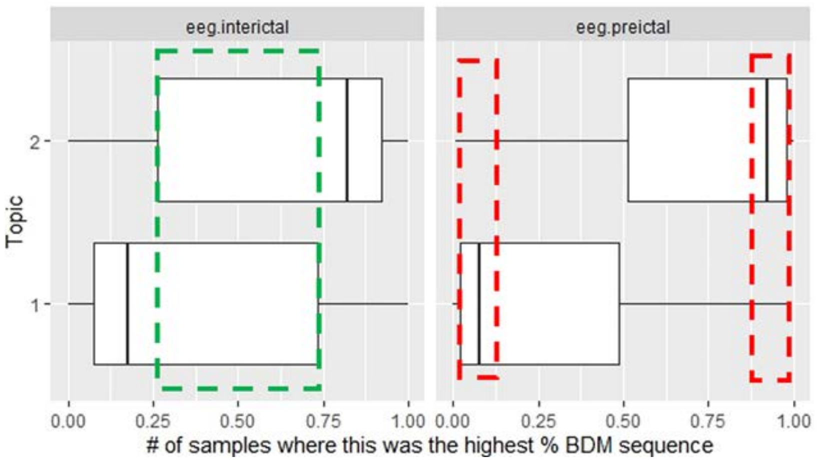


Figure 13. The difference between topic 1 and topic 2 as box plots.

A reason why the LDA approach works may be the similarity in the inner mechanics of this approach and algorithmic information dynamics (AID): both give high priority to generative processes. In the case of AID, this is the shortest computer program that can generate the object. The LDA modeling can also be seen as a generative process whereby the documents are created, so that we may infer, or reverse engineer, it. Documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over all the words.

As stated earlier, preictal and interictal subsets are the equivalents of groups of documents, BDM time series (based on EEG segments) represent documents, and BDM values (where digits are converted to letters) are the equivalent of words. Possible three-element permutations of 10 letters provide 1000 options, and adding two-element permutations (i.e., 100), we get a total of 1100. In fact, permutations that start with "0" can be excluded. Since the lower bound for a BDM value for a 16×16 matrix (e.g., an all-zero or all-one matrix) is 26.0067, values lower than this can also be excluded, getting 964 as the final number of permutations. For further data processing, the 1100 version continues to be used.

3.5 Final Version: Features Generated by Text Analysis Approach Fed to Neural Network/Multilayer Perceptron

The dataset provided by Kaggle [8] and described in Section 2 is highly unbalanced. For example, the dog-2 dataset, which is one of the largest among training sets, contains 500 interictal and only 42 preictal segments. To address this issue, the following new-data generation technique was used.

Based on the original BDM series, new ones were generated by adding elementwise to the value of the original BDM values a randomly generated number in the interval $[-1, 1]$. It was thought that these small changes would not distort the overall picture concerning a separate time series.

After converting BDM values from digits to letters and generation of text files, the final datasets to be used for ML consist of 500 interictal and 274 (of which 232 are replicas of the original TS or synthetically generated) preictal segments. The preictal and interictal subsets are tokenized and a two-topic LDA model generated according to the procedure described in the previous subsection. Further processing would include estimation of the beta-spread matrix for every segment. As an alternative, we could consider the terms that had the *greatest difference* in β between topic 1 and topic 2.

This can be estimated based on the log ratio of the two: $\log_2 \beta_1 / \beta_2$ (a log ratio is useful because it makes the difference symmetrical: β_1 being twice as large leads to a log ratio of 1, while β_2 being twice as large results in -1) [22].

To constrain it to a set of especially relevant words, we can filter for relatively common words, such as those that have a β greater than 1/1000 in at least one topic.

A general view of such a matrix is presented in Table 2. Theoretically, the maximum number of rows in such a matrix will not exceed 1100 (i.e., the maximum length of the dictionary). In practice, the number of words (i.e., unique BDM values converted to words) in a particular file (i.e., BDM-TS as text) ranges from 400 to 500 (or vocabulary length). For further research, a matrix template was generated with 1100 rows and two columns (the first column for the word and the second for the beta log ratio of this word).

A schematic view over data preprocessing steps is presented in Figure 14.

Word	Beta Log Ratio
...	...
bcg	−0.02116
bch	0
...	...
czh	0
czi	−1.28421
daa	−3.24629
dab	0.41629
...	...

Table 2. The beta log ratio matrix (a fragment). Zeros denote BDM unique values (as words) not present in the file that is analyzed.

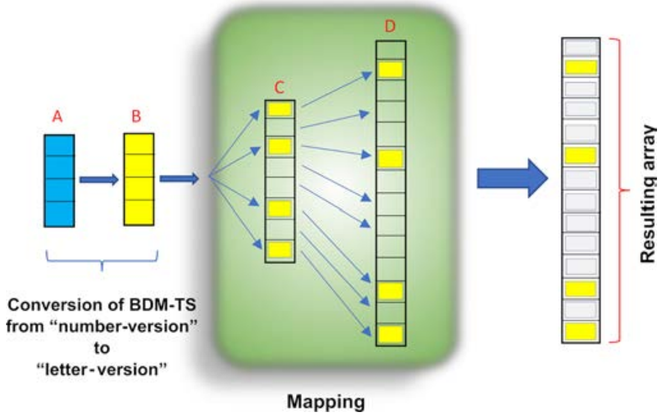

A
B
C
D
Conversion of BDM-TS from "number-version" to "letter-version"
Mapping
Resulting array

Figure 14. Data preprocessing steps: A, original/numeric BDM-TS; B, BDM-TS converted to words; C, vocabulary; D, dictionary, consisting of 1100 possible words. The resulting array has 1100 cells, and the cells for the words that are missing in the respective BDM-TS (gray color) are filled with zeros.

While generating LDA topics, the number of top terms in a topic was also set to the maximum value of 1100. Each matrix generated from a particular segment is mapped against this template, and the beta log ratio for the words existing in this segment is placed in the corresponding cells in the second column. For a missing word, the corresponding cell is filled with zeros.

Such matrices are generated for the preictal and interictal data subsets. Subsequently they are fed to NN as the training set, out of which 20 percent of data is reserved for validation.

The NN type used for this part of research is an MLP built on the H2O.ai platform. As the first step, the autoML approach is used. Based on the results, the first few models are then used for a grid search. After some parameter tuning, the best few models are trained based on an autoencoder built on a test dataset. The best models can finally be used for prediction. Table 3 presents an example of a confusion matrix of one of the models.

Using the preprocessing steps described earlier and feeding data to MLP, the results shown in Table 4 were obtained.

	0	1	Error Rate
0	311	79	0.202564 = 79 / 39
1	17	287	0.55921 = 17 / 304
Totals	328	366	0.138329 = 96 / 694

Table 3. An example of confusion matrix (vertical: actual; across: predicted) for F1-optimal threshold.

NN Type	Architecture	Data (EEG)	Perform (AUC)
Multilayer Perceptron (MLP)	4 layers total, 2 hidden layers with 50 units, Rectifier With-Dropout 0.5/0.5 Output Softmax	500 interictal 274 preictal 1000 test (autoencoder)	Training = 0.936 Validation = 0.937 5-fold CV = 0.874

Table 4. Results of building MLPs that are fed with pre-processed BDM-TS data. AUC—area under the curve.

3.6 Exploring and Processing Non-epileptic Data

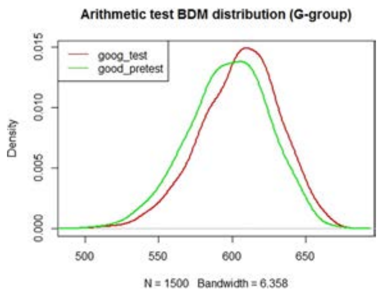
The datasets used here are available at [9].

In this experiment, all subjects are divided into two groups:

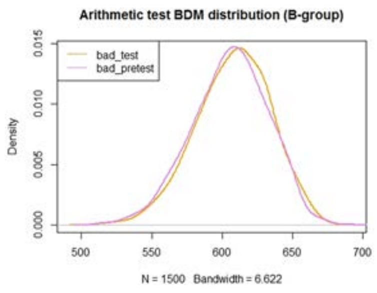
- Group G (or good-counters)—performing good quality count (mean number of operations per 4 minutes = 21, SD = 7.4).

- Group B (or bad-counters)—performing bad quality count (mean number of operations per 4 minutes = 7, SD = 3.6).

Since datasets are unbalanced (i.e., more subjects in group G) two equal-size groups of 10 subjects each were created by random sampling. After estimating BDM values for every file (pretest and during the test), the BDM value density distribution was estimated on mean per time step BDM values for each group (Figure 15) and both groups (Figure 16).



(a)



(b)

Figure 15. (a) BDM value distribution (pretest versus during the test) for the good-counters group. A shift of mean BDM to the right (or increased complexity) can be seen while performing the task. (b) BDM value distribution for the bad-counters group. The curves look quite similar.

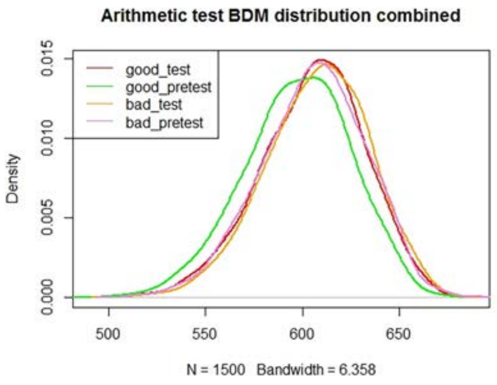


Figure 16. The BDM value distribution for both groups (i.e., good and bad counters).

A shift of complexity (by BDM) toward increased complexity can be noticed in the good-counters group (G-group) when performing the arithmetic task. In the bad-counters group (B-group), such a shift is almost missing. Analyzing the combined plot, it looks like there are noticeable changes in complexity within the G-group, while the B-group always shows a high complexity, comparable with the complexity of the G-group during the test.

The dataset generated by this research consists of 108 pretest subsets and 36 during the test subsets, which is of a too-limited volume for building an ML model. This was addressed by splitting every subset into four segments, with a resulting segment representing 15 seconds of the original EEG. On these short segments, BDM value was calculated and respective BDM-TSs generated. These final BDM-TSs have been converted to images, applying the GAF method. At this stage, the potential training set consisted of 142 images (one of the original subsets was excluded because of the nonstandard, i.e., less than 1-minute duration of the EEG record).

For a further increase in the number of images in the training set, the minimal information loss sparsification (MILS) algorithm [12] was used (Figure 17). This algorithm is one of the most recent tools in the AID field, designed to maximally preserve computable elements that contribute to the algorithmic information content of the data. Along with generating additional images for the training set, MILS also leads to data dimensionality reduction, which may be important when dealing with large datasets used for building ML models.

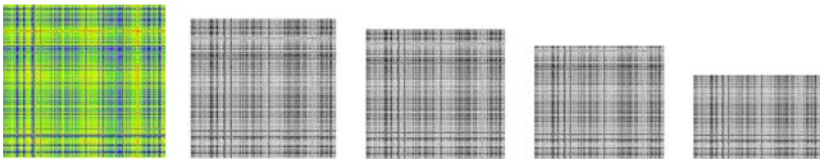


Figure 17. MILS applied to an image from the pretest set: on the left—original image, to the right—resulting images after applying MILS with number of reduced rows as follows: 5, 10, 20, 50 rows. Respective dimensionality reduction is: 46.2 KB (original image), 17.7 KB, 17.1 KB, 15.9 KB and 12.2 KB.

For the purpose of this research, the core of MILS was extracted from the R version available on GitHub [23] and integrated into the data processing pipeline. Using MILS, the final training set was upgraded to 808 images (i.e., 600 images in the pretest subset and 208 images in the during the test subset). This includes the original set and a few rounds of MILS with the number of rows by which the original image is reduced equivalent to 5, 10 and 20.

The training set was used to train a CNN model with a simple architecture that is close to an early version of LeNet proposed by Y. LeCun [24] with small adjustments suggested by [20]. Figure 18 presents this architecture.

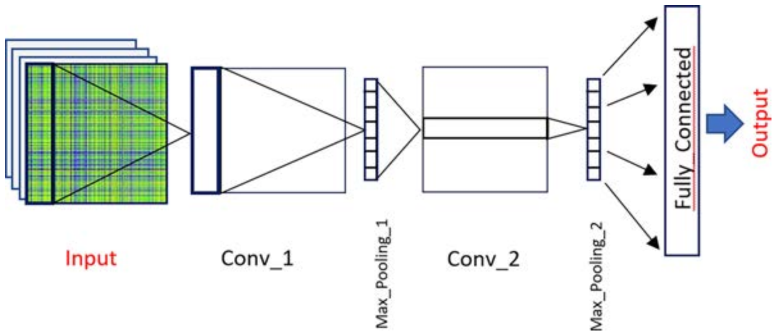


Figure 18. CNN architecture: Conv-1 is the first convolution layer with a 5×5 kernel, 20 filters and tanh activation function; Max_Pooling_1—first pooling layer with max pooling function, 2×2 kernel and 2×2 stride. Conv-2 is the second convolution layer with a 5×5 kernel, 50 filters and tanh activation function; Max_Pooling_2—second pooling layer with maximum pooling function, 2×2 kernel and 2×2 stride; fully connected—consists of two fully connected layers with 500 units and tanh activation in the first layer and two units for the output with softmax activation.

The validation accuracy achieved with this model was 0.79, which is quite low. It can be eventually increased by adjusting the kernel size or the number of filters, enlarging the volume of the training set and so on. Thus, the results in this paper are a preliminary lower bound on the potential best performance.

Since the scope of this research was to check the possibility of using NN for a BDM-TS classification task in principle, obtaining a state-of-the-art model is left for further research.

3.7 Limitations

Limitations to this research can be conventionally divided into several groups, the most relevant being connected with:

- Conceptual aspects (or using traditional statistical methods for tasks where algorithmic aspects are of primary importance)
- Data (i.e., data format, volume, etc.)
- Technical details

An important aspect to take care of when using tools based on statistical properties is the poor ability of these tools to preserve algorithmic information while processing the data. This is described in more detail in [5, 16].

Depending on the method that is used to transform the data, its intermediary and final format are different (e.g., time series, arrays and images) and can bear the shortcomings as well as the advantages of the respective methods. This research uses combinations of statistical and algorithmic tools, and it may be difficult to monitor the dynamics of their mutual influence. This is especially valid when feeding data that is originally of an algorithmic nature (i.e., BDM-TS) to ML models (which are mostly based on statistical principles).

A serious limitation is the volume of data, especially their non-epileptic portion and especially for building ML models. New data generation techniques address this issue only partially.

Technical limitations generally relate to a specific NN architecture/model. An example of technical details that depend on the tools that are used in this research would be text mining tools: one of the steps in text data processing is removing stopwords (stopwords are words you do not want to track, like "he," "hi" or "e.g."). In a BDM-TS, after conversion to words such words may be present (e.g., the word "big" that encodes BDM = 297 in the dog-2 set is present 1485 times). Using a regular approach, these words are removed and this will distort the picture.

4. Conclusion and Final Notes

A number of tools used in this research are of a statistical nature. Many of these tools are based on statistical properties (e.g., statistically driven pattern recognition), and with them there is a risk of missing any nonlinear and algorithmic regularity if it does not show a statistical property [12]. Because of this, applying such methods to BDM value over time (BDM-TS) is probably not the best idea and should be performed with the required amount of care.

There are tools of a more or less pure algorithmic information dynamics (AID) nature (e.g., minimal information loss sparsification (MILS) [23, 24] and online algorithmic complexity calculator (OACC) [15]), but the set of such tools is of a still-limited size. In many instances, the current research was an attempt at using available tools (including the ones appropriate for time series analysis) to get some intuition concerning the dynamics of the complexity of an object of interest, which in this framework is the brain, and try using the result to infer some conclusions.

One of the behind-the-scene concepts for this research is the concept of spikes and sharp waves as forerunners of an epileptic state (i.e., preictal period). Once present on electroencephalography (EEG), we can expect they should be captured by BDM-TS generated by this EEG as well. Taking into account that the duration of a spike is approximately 80 milliseconds and of a sharp wave 200 milliseconds, and the time step of the BDM-TS for the dog subjects (with EEG resolution of 400 Hz) is 25 milliseconds, there is a chance for spikes and sharp waves to be present on BDM-TS as distinct events. For the human subjects, where the EEG resolution in this dataset is 5000 Hz, the BDM-TS is much more fine-grained, with duration of the time step of only 2 milliseconds; the chance for detecting such events of interest might be higher.

For an eventual clinical application, the machine learning (ML) models described here evidently need further refinement, but some aspects concerning BDM-TS revealed by this research can pave the way for doing this as well as serve as possible avenues for future research.

This paper is mostly focused on what can be done and presents some ways to do it. The question of how it works is mostly beyond the scope of the research and can serve as quite a captivating exploratory field for future projects.

And finally, we would like to believe that the results of this research give some insights and contour some directions in using BDM in particular and AID concepts and tools in general for more knowledgeable decision-making in a clinical and paraclinical setting.

To demonstrate the main steps of data processing, including classifying a certain clip from the dog-2 set (i.e., interictal or preictal), a Shiny application was built, described in the Appendix.

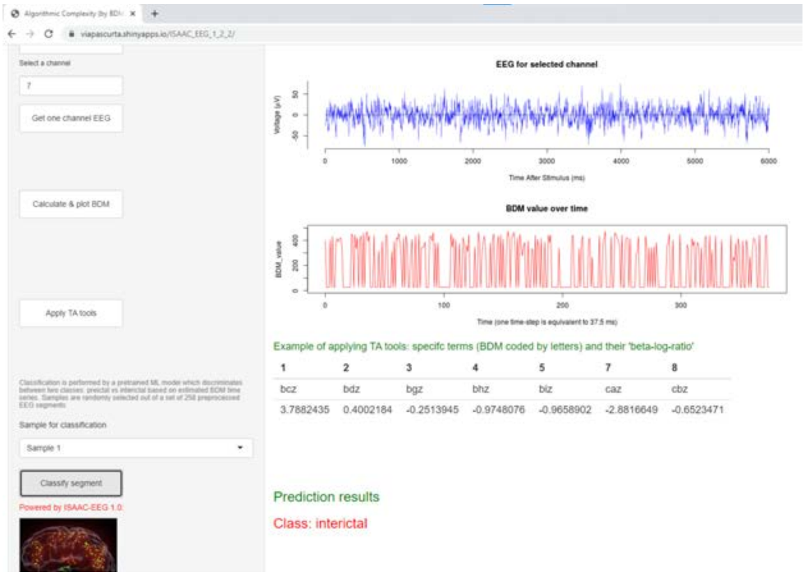
A. Appendix

The algorithmic complexity (by BDM) EEG Analyzer is available at viapascurta.shinyapps.io/ISAAC_EEG_1_2_2.

This is a demo Shiny (R) application for the analysis of an EEG clip, including estimation of its algorithmic (Kolmogorov) complexity by BDM. The application is built using concepts and tools described in the current paper.

The application accepts EEG clips as .mat files provided by the Kaggle competition [8] form, where such a file can be downloaded after registration. A sample file to play with is available at
www.dropbox.com/s/ou6bcjk4rkcenbd/Dog_2_interictal_segment_
0003.mat?dl=0.

The first step in running the application is importing/loading the file (it should be on the user's computer). By clicking the "Convert to csv" button, the file is converted to .csv, and it should be saved on the user's computer. Then the .csv file is loaded and a number of possibilities become available: (*a*) displaying the 16-channel EEG for a selected subsegment; (*b*) separately visualizing any of the 16 channels; (*c*) calculating and displaying the BDM-TS for the selected subsegment; and (*d*) applying text analysis tools for displaying the first seven beta log ratios for the respective subsegments.



Classification is performed by a pretrained ML model which discriminates between two classes: preictal vs interictal based on estimated BDM time series. Samples are randomly selected out of a set of 258 preprocessed EEG segments

Example of applying TA tools: specifc terms (BDM coded by letters) and their 'beta-log-ratio'

1	2	3	4	5	7	8
bcz	bdz	bgz	bhz	biz	caz	cbz
3.7882435	0.4002184	-0.2513945	-0.9748076	-0.9658902	-2.8816649	-0.6523471

Prediction results

Class: interictal

Figure A.1. The GUI of the algorithmic complexity EEG analyzer.

And finally, the classification task can be performed by selecting a sample for classification. The application randomly selects a sample out of 258 preprocessed samples. The result will be shown on the main panel under "Prediction results" after clicking the "Classify segment" button.

References

[1] N. Kamel and A. Saeed Malik, eds., *EEG/ERP Analysis: Methods and Applications*, Boca Raton, New York: CRC Press/Taylor & Francis Group, 2015.

[2] D. Puthankattil Subha, P. K. Joseph, R. Acharya U and C. M. Lim, "EEG Signal Analysis: A Survey," *Journal of Medical Systems*, **34**(2), 2010 pp. 195–212. doi:10.1007/s10916-008-9231-z.

[3] J. D. Simeral, T. Hosman, J. Saab, S. N. Flesher, M. Vilela, B. Franco, J. N. Kelemen, et al., "Home Use of a Percutaneous Wireless Intracortical Brain-Computer Interface by Individuals with Tetraplegia," *IEEE Transactions on Biomedical Engineering*, **68**(7), 2021 pp. 2313–2325. doi:10.1109/TBME.2021.3069119.

[4] G. Rodriguez-Bermudez and P. J. Garcia-Laencina, "Analysis of EEG Signals Using Nonlinear Dynamics and Chaos: A Review," *Applied Mathematics and Information Sciences*, **9**(5), 2015 pp. 2309–2321. www.naturalspublishing.com/files/published/d4c96j5fp99493.pdf.

[5] H. Zenil, “A Review of Methods for Estimating Algorithmic Complexity: Options, Challenges, and New Directions,” *Entropy*, **22**(6), 2020 612. doi:10.3390/e22060612.

[6] J. N. Y. Franklin and C. P. Porter, “Key Developments in Algorithmic Randomness.” arxiv.org/abs/2004.02851.

[7] H. Zenil, “Towards Demystifying Shannon Entropy, Lossless Compression and Approaches to Statistical Machine Learning,” *Proceedings*, **47**(1), 2020 24. doi:10.3390/proceedings2020047024.

[8] American Epilepsy Society Seizure Prediction Challenge, 2014. www.kaggle.com/c/seizure-prediction.

[9] I. Zyma, S. Tukaev, I. Seleznov, K. Kiyono, A. Popov, M. Chernykh and O. Shpenkov, “Electroencephalograms during Mental Arithmetic Task Performance,” *Data*, **4**(1), 2019 14. doi:10.3390/data4010014.

[10] H. Zenil and N. Kiani, instrs., “Algorithmic Information Dynamics: A Computational Approach to Causality and Living Systems from Networks to Cells,” MOOC by Complexity Explorer, Santa Fe Institute (Jun 12, 2018 to Oct 13, 2018), Santa Fe, NM. www.complexityexplorer.org/courses/63-algorithmic-informationdynamics-a-computational-approach-to-causality-and-livingsystems-from-networks-to-cells-2018.

[11] H. Zenil, S. Hernández-Orozco, N. A. Kiani, F. Soler-Toscano, A. Rueda-Toicen and J. Tegnér, “A Decomposition Method for Global Evaluation of Shannon Entropy and Local Estimations of Algorithmic Complexity,” *Entropy*, **20**(8), 2018 605. doi:10.3390/e20080605.

[12] H. Zenil, N. A. Kiani, F. Marabita, Y. Deng, S. Elias, A. Schmidt, G. Ball and J. Tegnér, “An Algorithmic Information Calculus for Causal Discovery and Reprogramming Systems,” *iScience*, **19**, 2019 pp. 1160–1172. doi:10.1016/j.isci.2019.07.043.

[13] F. Soler-Toscano, H. Zenil, J.-P. Delahaye and N. Gauvrit, “Calculating Kolmogorov Complexity from the Output Frequency Distributions of Small Turing Machines,” *PLoS ONE*, **9**(5), 2014 e96233. doi:10.1371/journal.pone.0096223.

[14] J.-P. Delahaye and H. Zenil, “Numerical Evaluation of Algorithmic Complexity for Short Strings: A Glance into the Innermost Structure of Randomness,” *Applied Mathematics and Computation*, **219**(1), 2012 pp. 63–77. doi:10.1016/j.amc.2011.10.006.

[15] H. Zenil, F. Soler-Toscano, N. Gauvrit, N. A. Kiani, H. Singmann, G. Scott, S. Hernández, et al., “The Online Algorithmic Complexity Calculator (OACC) v3.0,” Algorithmic Dynamics Lab, Science for Life Laboratory (SciLifeLab), Unit of Computational Medicine, Center for Molecular Medicine at the Karolinska Institute in Stockholm, Sweden 2018. www.algorithmicdynamics.net/software.html.

[16] V. Iapascurta, "Block Decomposition Method and Traditional Machine Learning for Epileptic Seizure Prediction," presentation given at *26th IFIP WG 1.5 International Workshop AUTOMATA 2020, Special Session on Algorithmic Information Dynamics, August 10–12*, Stockholm, Sweden 2020. www.automata2020.com/videos--material.html.

[17] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar and P.-A. Muller, "Deep Learning for Time Series Classification: A Review," *Data Mining and Knowledge Discovery*, **33**(4), 2019 pp. 917–963. doi:10.1007/s10618-019-00619-1.

[18] E. W. Weisstein. "Recurrence Plot" from Wolfram MathWorld—A Wolfram Web Resource. mathworld.wolfram.com/RecurrencePlot.html.

[19] N. Marwan, C. Romano, M. Thiel and J. Kurths, "Recurrence Plots for the Analysis of Complex Systems," *Physics Reports*, **438**(5–6), 2007 pp. 237–329. doi:10.1016/j.physrep.2006.11.001.

[20] R. Damasevicius, R. Maskeliunas, M. Wozniak and D. Połap, "Visualization of Physiologic Signals Based on Hjorth Parameters and Gramian Angular Fields," *IEEE 16th World Symposium on Applied Machine Intelligence and Informatics (SAMI 2018)*, Kosice and Herlany, Slovakia, 2018, Piscataway, NJ: IEEE 2018. doi:10.1109/SAMI.2018.8323992.

[21] N. Hatami, Y. Gavet and J. Debayle, "Classification of Time-Series Images Using Deep Convolutional Neural Networks." arxiv.org/abs/1710.00886v2.

[22] J. Silge and D. Robinson, *Text Mining with R: A Tidy Approach*, Boston: O'Reilly, 2017.

[23] algorithmicnaturelab/MILS. (Oct 12, 2022) github.com/algorithmicnaturelab/MILS.

[24] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," in *Proceedings of the IEEE*, **86**(11), 1998 pp. 2278–2324. doi:10.1109/5.726791.

[25] H. Zenil, N. A. Kiani, A. Rueda-Toicen, A. A. Zea and J. Tegnér, "Data Dimension Reduction and Network Sparsification Based on Minimal Algorithmic Information Loss." arxiv.org/abs/1802.05843v6.