Feature extraction from electroencephalographic records using EEGFrame framework

Alan Jović[*], Lea Suć[*], and Nikola Bogunović[*]

[*] Faculty of Electrical Engineering and Computing, University of Zagreb / Department of Electronics, Microelectronics, Computer and Intelligent Systems, Unska 3, 10 000 Zagreb, Croatia

{alan.jovic, lea.suc, nikola.bogunovic}@fer.hr

Abstract - Analysis of electroencephalographic (EEG) signals usually includes visual inspection of the signal, feature extraction, and model generation. Computer-aided nonlinear feature extraction from EEG in particular has already led to improved descriptive and prognostic models of brain states and disorders. However, in this field, there is a lack of freely available powerful tools for scientific exploration of EEG that would help researchers to compare the results of their work with others. Especially, because of the great diversity of the proposed methods for EEG analysis, there exists a need for a joint framework for inspection, extraction and visualization performed on the EEG records. The aim of this paper is to introduce such a framework, called EEGFrame, with its implementation in Java. The framework currently supports the analysis of standard EDF records via signal inspection, feature extraction, and feature vectors storage for knowledge discovery. EEGFrame is the result of refactoring and extension of the HRVFrame framework for heart rate variability analysis, with added methods for EEG analysis. This paper describes the properties and capabilities of the framework and discusses its relevance with respect to similar work. The main advantage of EEGFrame is its support for numerous linear and nonlinear methods described in literature.

I. Introduction

Computer models of brain functioning have become increasingly complex in recent years. The extraction of nonlinear features of neurological time-series, in particular, electroencephalography (EEG) and magnetoencephalography (MEG) enables ever more accurate analysis of diverse brain disorders and states. The analysis of EEG/MEG using nonlinear features aids medical personnel in better understanding of the underlying physiological aspects of the disorder as well as for classification of patients and prediction of disorder onset [1]. Typical applications include modeling of disorders such as epilepsy, coma, schizophrenia, Alzheimer's disease, Parkinson's disease, etc., while brain functioning during normal or altered conscious states include normal wake state, sleep, relaxation and meditation [1,2]. Complexity of the time-series is particularly pronounced under epileptic seizures, and this area of research usually involves creating accurate models for early detection of possible seizure [2].

Most of the features used for EEG/MEG analysis are nonlinear due to the inherent nonlinear and non-stationary behavior of the time-series. There are a few exceptions, mostly covering basic statistical properties of the series. Due to a large number of methods available for studying variability of biomedical time-series [3], computer-aided analysis of EEG/MEG relies on availability of tools for feature extraction and data mining. While there already exist several tools for such analyses, there are very few open-source frameworks that allow researchers to experiment with novel models of disorders. Currently, there are a few initiatives that try to standardize data inputs and outputs. Most of them are based on Matlab and/or C/C++ [4,5]. A framework in Python (PyEEG) was recently developed that tries to bridge the gap between EEG data input and data mining [6]. To our knowledge, there is no extensive stand-alone open-source framework that would cover the majority of features employed in EEG analysis, while at the same time enabling data input, feature extraction, EEG visualization, and storing feature vectors in a format suitable for data mining.

The aim of this paper is to introduce and describe such a framework, which we call EEGFrame. The framework is written in Java as a stand-alone application, but with embedding possibility, and is intended for use by both medical practitioners as well as researchers in the field of biomedical engineering. EEGFrame has been developed as an offspring of an already existing framework, HRVFrame that is used for heart rate variability analysis [7]. The extension includes visualization of the EEG signals loaded from a standard EDF (European Data Format) record, extraction of a number of nonlinear features applicable to EEG, and adaptation of frequency domain features to cover typical EEG frequency bands.

The paper is organized as follows. In Section 2, we show an overview of the framework, with description of input and output data formats. Section 3 describes the features that are implemented in the framework. Section 4 describes the visualization of the EEG records. Comparison to similar work and discussion are shown in section 5. Conclusion is given in section 6.

II. Overview of the Framework

A. *System overview, input and output*

The main purpose of the EEGFrame framework is to extract feature vectors from EEG records and store them for further knowledge discovery. The framework currently

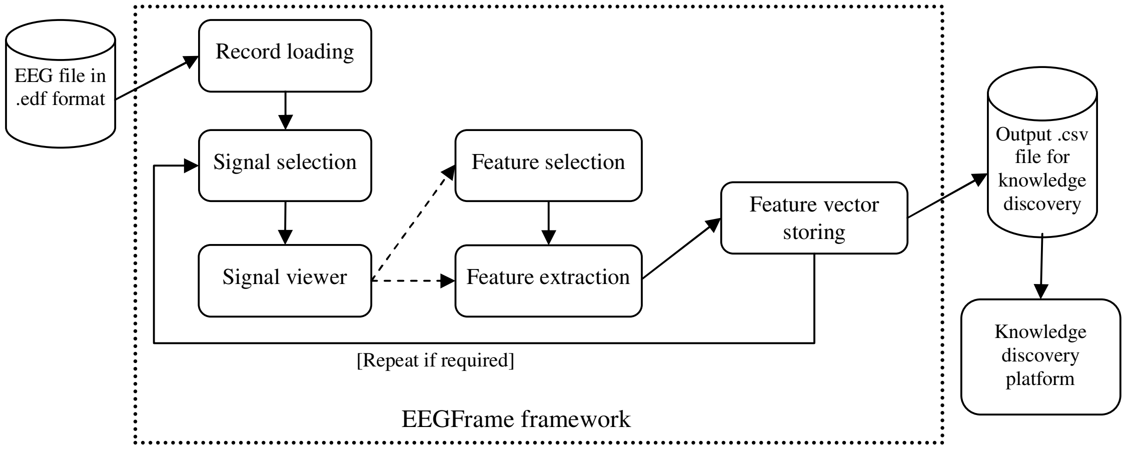


Fig. 1. Overview of the EEGFrame framework

supports extraction from standard EDF records with plans to support EDF+ standard in the future. Additionally, the framework enables visual inspection of each EEG record with numerous display options to suit the researcher's needs. Transformation of signal data from .edf file to a file in textual format is also possible through the framework. The overview of the system is shown in Fig. 1.

The framework assumes that the data in EDF format is available, and that it has been filtered (the framework does not include EEG preprocessing methods). It then enables visual inspection of a single record and feature extraction from the record, by specifying the extraction parameters. The output file is recorded in .csv format that can be read by most of the open-source knowledge discovery platforms such as Weka [8] or Rapidminer [9]. The output file contains feature vectors that contain values for all the features that were selected for the analysis. Each new feature vector is appended as a row to the end of the file if the analysis involves multiple segments. The user can always create a new output file. The header of the output file contains a list of all the possible features implemented in the framework. For those features that are not extracted as a part of the feature vector, the symbol "?" is added.

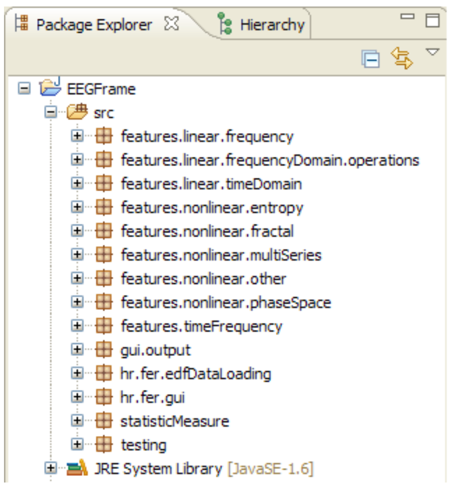
B. Framework's internal structure

EEGFrame consists of several Java packages, Fig. 2. Basically, there are two types of packages. There are the packages that contain all of the classes intended for data transformation and feature extraction. These packages start with the label "features". There are also other auxiliary packages that consist of classes used for constructing graphical user interface and record visualization, data input, data output, and testing of the framework. The package "statisticMeasure" is essential for functioning of some of the feature extraction classes. Likewise, the package "features.linear.frequencyDomain. operations" contains several classes needed for frequency domain analysis of EEG records (e.g. complex numbers, fast Fourier transform).

The packages consist of public classes with well-documented and thoroughly tested methods. All of the implemented classes with feature extraction methods provide reference about the source document(s) that describe the mathematical background and rationale for the implemented methods.

III. FEATURE EXTRACTION METHODS

The framework currently supports extraction of roughly 50 time-domain, frequency domain, time-frequency and nonlinear features. In Table I, a summary report of the implemented methods is provided. The reference next to the name of the method refers to the literature from which the method was implemented. If a method allows extraction of several features, a list of the features is also included in Table I. If a feature is parametric, the parameters that need to be provided are shown. Some of the implemented nonlinear methods are used for measuring a common or combined property of two or more time-series (e.g. mutual dimension, synchronization likelihood). We have included these methods in a separate package, "features.nonlinear.multiSeries", as to distinguish them from the other nonlinear methods that can be applied to only a single time-series (i.e. single EEG electrode signal).



```
EEGFrame
  src
    features.linear.frequency
    features.linear.frequencyDomain.operations
    features.linear.timeDomain
    features.nonlinear.entropy
    features.nonlinear.fractal
    features.nonlinear.multiSeries
    features.nonlinear.other
    features.nonlinear.phaseSpace
    features.timeFrequency
    gui.output
    hr.fer.edfDataLoading
    hr.fer.gui
    statisticMeasure
    testing
  JRE System Library [JavaSE-1.6]
```

Fig. 2. The structure of the framework

TABLE I. METHODS AND FEATURES IMPLEMENTED IN THE FRAMEWORK

Method [reference]	List of features / parameters	Package	Class
Mean [10]	Single feature	features.linear.timeDomain	Mean
Standard deviation [10]	Single feature	features.linear.timeDomain	StandardDeviation
Fano factor [11]	Single feature	features.linear.timeDomain	FanoFactor
Autocorrelation coefficient [11]	Single feature	features.linear.timeDomain	Autocorrelation-Coefficient
Mean of absolute values of first differences [10]	Single feature	features.linear.timeDomain	MeanOfAbsoluteValues OfFirstDiff
Mean of absolute values of second differences [10]	Single feature	features.linear.timeDomain	MeanOfAbsoluteValues OfSecondDiff
Mean of absolute values of first differences normalized [10]	Single feature	features.linear.timeDomain	MeanOfAbsoluteValues OfFirstDiffNormalized
Mean of absolute values of second differences normalized [10]	Single feature	features.linear.timeDomain	MeanOfAbsoluteValues OfSecondDiffNormalize d
Spectral analysis [10]	AlfaPSD, BetaPSD, GamaPSD, DeltaPSD, ThetaPSD (5 feat.) / sample frequency, FFT PSD (window) or Burg PSD estimate (AR model order)	features.linear.frequency	SpectralAnalysis
Spectral entropy [12]	Single feature / lower and upper frequency limit, FFT PSD or Burg PSD estimate	features.linear.frequency	SpectralEntropy
Approximate entropy [13]	Single feature / m factor, r	features.nonlinear.entropy	ApEn
Maximum approximate entropy [13]	MaxApEn, r for MaxApEn / *m* factor	features.nonlinear.entropy	ApEn
Carnap entropy 1D [14]	Single feature	features.nonlinear.entropy	CarnapEntropy1D
Corrected conditional Shannon entropy [15]	Single feature / dimension, no. of bins	features.nonlinear.entropy	CorrectedConditional-SnannonEntropy
Rényi entropy [16]	Single feature / order	features.nonlinear.entropy	RenyiEntropy
Sample entropy [17]	Single feature / m factor, r	features.nonlinear.entropy	SampEn
Maximum sample entropy [17]	MaxSampEn, r for MaxSampEn / *m* factor	features.nonlinear.entropy	SampEn
Detrended fluctuation analysis [18]	DFAAlphaS, DFAAlphaL (2 feat.) / minimum analyzed segment length, bound for AlphaL, long range calculation flag	features.nonlinear.fractal	DFA
Higuchi's fractal dimension [19]	Single feature / *kmax*	features.nonlinear.fractal	HiguchiDimension
Hurst exponent [11]	Single feature	features.nonlinear.fractal	HurstExponent
Cross recurrence [20]	CRPRecurrence rate, CRPLmean, CRPDET, CRPSh. ent. rec., laminarity (5 feat.) / probe, dimension,lag, r	features.nonlinear. multiSeries	CrossRecurrence
Mutual dimension [21]	Single feature / dimension 1, dimension 2, lag 1, lag 2, no. of bins	features.nonlinear. multiSeries	MutualDimension
Synchronization likelihood [22]	Single feature / signal indices, dimension, lag, no. of bins, recurrence number, ro	features.nonlinear. multiSeries	SynchronizationLikeliho od
Allan factor [11]	Single feature / observational window	features.nonlinear.other	AllanFactor
Lempel-Ziv complexity [23]	Single feature	features.nonlinear.other	LempelZivComplexity
Nonlinear forecasting [24]	Single feature / dimension, lag	features.nonliner.other	NonlinearForecasting
Correlation dimension [25]	Single feature / dimension, lag, no. of bins	features.nonlinear. phaseSpace	CorrelationDimension
Central tendency measure [26]	Single feature / dimension, lag, r	features.nonlinear. phaseSpace	CTM
Largest Lyapunov exponent [27]	Single feature / dimension, trajectory length	features.nonlinear. phaseSpace	LyapunovExponent
Recurrence plot [28]	AVG number of neighbors, recurrence rate, Lmean, DET, Sh. ent. rec., laminarity (6 feat.) / dimension, lag, r	features.nonlinear. phaseSpace	RecurrencePlot
Spatial filling index [29]	Single feature / dimension, lag, no. of bins	features.nonlinear. phaseSpace	SpatialFillingIndex
Standard deviations ratio [30]	Single feature	features.nonlinear. phaseSpace	StandardDeviationsRatio
Haar wavelet standard deviation [11]	Single feature / scale	features.timeFrequency	HaarWaveletStandard Deviation
HilbertHuangTransform [31]	Instantaneous frequencies (IF), amplitudes, intrinsic mode functions, max IF, amplitudes for max IF (multiple features) / sampling period	features.timeFrequency	HilbertHuangTransform

IV. EEG SIGNAL VISUALIZATION

It is important for researchers to have the possibility of visual inspection of EEG records. The standard EDF format supports the International 10-20 system of electrode placements [32]. In this EEG system, it is possible to have 21 or more electrodes that measure the voltages of the brain's micro-currents. The visualization of the signals recorded by such a system can be performed in many ways. The standard display is a rectangular grid with each rectangle covering exactly 0.2 s times 50 μV. Since the number of signals can vary, and not all of them are always necessary to include for certain disorders, the visualization part includes the options to display only specific electrode signals as well as all signals. Additionally, several time and amplitude scales can be selected, and the user can easily browse through the entire record. An example of EEG visualization is shown in Fig. 3.

The graphical user interface also allows users to display the properties of the records (e.g. date taken, start time, length in seconds, number of signals, header length in bytes), that are available from the .edf file header.

V. COMPARISON TO SIMILAR WORK AND DISCUSSION

In Table II, we compare the currently most popular open-source software in EEG analysis community with EEGFrame. There are several open-source EEG analysis software tools available today. Most of the software is based on Matlab/Octave or C/C++, and only a few of them are specialized for EEG analysis. One of the problems with Matlab based software is that its use depends on whether the user has a Matlab license. This can be solved to an extent by using Octave instead of Matlab (e.g. project BioSig [5]). The other problem is that the software is not monolithic or cohesive – i.e. not all of the required methods can be easily found and used. This severely limits the usefulness and user-friendliness of the large Matlab-based software toolboxes such as EEGLab [4]. The main advantage is the large community support and extensions that Matlab provides. This includes signal analysis and transformations, feature extraction, classification, as well as visualization capabilities.

While no software or framework today contains all of the methods that the researchers used in EEG analysis, it can be safely concluded that two of the frameworks: PyEEG, written in Python, and EEGFrame, written in Java, provide the largest number of individual feature extraction methods. Both of the frameworks focus on nonlinear features. The advantage of EEGFrame, when compared with PyEEG lies both in the larger number of currently implemented individual features, as well as in visualization and data input/output capability. PyEEG is a framework that contains only the feature extraction methods; it relies heavily on other software (e.g. BioSig) for particular application [6].

The principle goal of the researchers in EEG analysis is to acquire the best model for a particular brain state or disorder. In order to claim that their model is the best possible one within the limits of current human knowledge, the researchers need to be able to efficiently compare their results, particularly, their features or features combinations [33], with the work of others. EEGFrame facilitates this comparison because it provides the researcher with a large number of features' implementations that other researchers applied in EEG

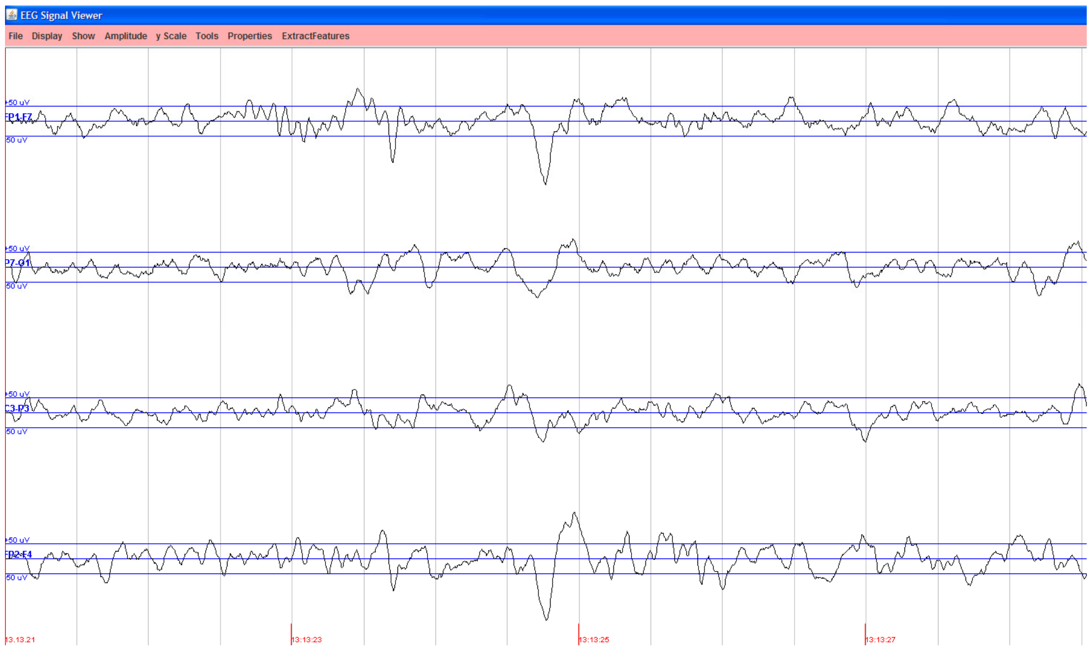


Fig. 3. Visualization of the EEG record chb01_27.edf from Physionet CHB-MIT database for electrodes FP1-F7, P7-O1, C3-P3, and FP2-F4. The user can select any number of signals present in the record for display

TABLE II. OPEN-SOURCE EEG ANALYSIS SOFTWARE COMPARISON

Software	Purpose	Implementation language	Type	Implemented features
EEGLab [4]	Extensive Matlab toolbox for EEG analysis: visualization, 3D brain modelling, feature extraction, several plugins (NFT, ERICA, BCILAB...)	Matlab	Embedded	Time/frequency/time-frequency/independent component transformations and features / unknown total number of features
BioSig [5]	Reading and writing routines for many biomedical time-series data formats; EEG preprocessing, visualization, feature extraction (multivariate autoregressive modeling) and classification (via Matlab/Octave)	C/C++, Matlab (or Octave)	Some functions standalone, mostly embedded	Time/frequency/time-frequency transformations and features, unknown total number of features
PyEEG [6]	Feature extraction framework, feature vector output for data mining	Python	Embedded	Frequency/nonlinear features, currently 21 features in total
EEGFrame	Signal inspection, feature extraction framework, handles .EDF input, feature vector output for data mining	Java	Stand-alone or embedded	Time/frequency/time-frequency/nonlinear features, currently 49 features in total

analysis. Hence, with the assumption of using the same dataset, a researcher can claim that his feature or features' combination is superior to the features employed by others for a particular problem.

Batch feature extraction from multiple EEG records is currently not possible through the EEGFrame graphical user interface. However, adapting the framework for batch extraction from several records is not difficult and is planned for future versions. Adding more methods to EEGFrame, especially other nonlinear ones (e.g. phase synchronization, Hjorth mobility and complexity, SVD entropy, Fisher information, etc.) is also planned.

EEGFrame can be employed both as a stand-alone product, and as a part of a larger system. Herein, only the feature packages would be integrated with other software. The framework is written entirely in Java, which alleviates the problems with platform dependencies. Thus, EEGFrame can be used on any operating system, provided that the Java virtual machine (version 5.0+) is installed on the system. EEGFrame is available as open-source, GPL 2 licensed software for non-commercial biomedical time-series analysis applications from the following web site: http://www.zemris.fer.hr/~ajovic/eegframe/eegframe.html.

VI. CONCLUSION

This paper presented a novel feature extraction framework from EEG that is implemented in Java. The framework currently implements many features used by researchers in the domain of EEG analysis and thus enables easier comparison of academic work. It can be used both as a part of another, larger system, or as a stand-alone EEG visualization and analysis application.

For future work, the framework is planned to be updated with additional nonlinear features. Also, visualization of phase space features as well as batch analysis of EEG records is in order.

REFERENCES

[1] C. J. Stam, "Nonlinear dynamical analysis of EEG and MEG: Review of an emerging field," Clin. Neurophysiol., vol. 116, pp. 2266–2301, Oct. 2005.

[2] K. Lehnertz, "Epilepsy and Nonlinear Dynamics," J. Biol. Phys., vol. 34, pp. 253–266, Aug. 2008.

[3] A. Bravi, A. Longtin, and A. J. E. Seely, "Review and classification of variability analysis techniques with clinical applications," BioMed. Eng. OnLine, vol. 10, p. 90, Oct. 2011.

[4] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics", J. Neurosci. Methods, vol. 134, pp. 9–21, Mar. 2004.

[5] C. Vidaurre, T. H. Sander, and A. Schlögl, "BioSig: The Free and Open Source Software Library for Biomedical Signal Processing," Comput. Intell. Neurosci., p. 935364, 2011.

[6] F. S. Bao, X. Liu, and C. Zhang, "PyEEG: An open source Python module for EEG/MEG feature extraction," Comput. Intell. Neurosci., p. 406391, 2011.

[7] A. Jovic and N. Bogunovic, "HRVFrame: Java-Based Framework for Feature Extraction from Cardiac Rhythm," Lecture Notes in Artificial Intelligence, vol. 6747, pp. 96–100, Jul. 2011.

[8] I. H. Witten and E. Frank, Data mining: Practical machine learning tools and techniques, 3rd ed., San Francisco: Morgan Kaufmann, 2011.

[9] (2012) RapidMiner. [Online]. Available: http://rapid-i.com/content/view/181/190/

[10] X.-W. Wang, D. Nie, and B.-L. Lu, "EEG-Based Emotion Recognition Using Frequency Domain Features and Support Vector Machines," Lecture Notes in Computer Science, vol. 7062, pp. 734–743, 2011.

[11] M. C. Teich, S. B. Lowen, B. M. Jost, K. Vibe-Rheymer, and C. Heneghan, "Heart-Rate Variability: Measures and Models," in Dynamic Analysis and Modeling, ser. Nonlinear Biomedical Signal Processing, M. Akay, Ed. New York: IEEE Press, 2001, vol. II, ch. 6, pp. 159–213.

[12] I. A. Rezek and S. J. Roberts, "Stochastic Complexity Measures for Physiological Signal Analysis," IEEE Trans. Biomed. Eng., vol. 45, no. 9, pp. 1186–1191, Sep. 1998.

[13] S. M. Pincus and A. L. Goldberger, "Physiological time-series analysis: what does regularity quantify?" Am. J. Physiol., vol. 266, no. 4, (Heart Circ. Physiol., vol. 35), pp. H1643–H1656, Apr. 1994.

[14] F. Jovic, D. Krmpotic, and A. Jovic, "Process entropy and informational macrodynamics in a ceramic tile plant," in Proc. 32nd Croatian Society for Information and Communication Technology, Electronics and Microelectronics - MIPRO, vol. III, pp. 50–53, 2009.

[15] A. Porta, G. Baselli, D. Liberati, N. Montano, C. Cogliati, T. Gnecchi-Ruscone, et al., "Measuring regularity by means of a corrected conditional entropy in sympathetic outflow," Biol. Cybern., vol. 78, no. 1, pp. 71–78, 1998.

[16] K. Waheed and F. M. Salam, "A Data-Derived Quadratic Independence Measure for Adaptive Blind Source Recovery in Practical Applications," in Proc. 45th IEEE Int. Midwest Symposium on Circuits and Systems, pp. 473–476, 2002.

[17] J. S. Richman and J. R. Moorman, "Physiological time-series analysis using approximate entropy and sample entropy," Am. J.

Physiol. (Heart Circ. Physiol.), vol. 278, no. 6, pp. 2039–2049, Jun 2000.

[18] C.-K. Peng, S. Havlin, H. E. Stanley, and A. L. Goldberger, "Quantification of scaling exponents and crossover phenomena in nonstationary heartbeat time series," Chaos, Solitons, & Fractals, vol. 5, no. 1, pp. 82–87, Jan. 1995.

[19] T. Higuchi, "Approach to an irregular time series on the basis of the fractal theory", Physica D, vol. 31, issue 2, pp. 277--283, Jun. 1988.

[20] N. Marwan, M. C. Romano, M. Thiel, and J. Kurths, "Recurrence plots for the analysis of complex systems," Physics Reports, vol. 438, issues 5–6, pp. 237–329, Jan. 2007.

[21] C. J. Stam, T. C. van Woerkom, W. S. Pritchard, "Use of non-linear EEG measures to characterize EEG changes during mental activity", Electroencephalogr. Clin. Neurophysiol., vol. 99, no. 3, pp. 214–224, Sep. 1996.

[22] C. J. Stam, B. W. van Dijk, "Synchronization likelihood: an unbiased measure of generalized synchronization in multivariate data sets", Physica D: Nonlinear Phenomena, vol. 163, issues 3–4, pp. 236–251, Mar. 2002.

[23] X.-S. Zhang, Y.-S. Zhu, N. V. Thakor, and Z.-Z. Wang, "Detecting Ventricular tachycardia and fibrillation by complexity measure," IEEE Trans. Biomed. Eng., vol. 46, no. 5, pp. 548–555, May 1999.

[24] G. Sugihara and R. M. May, "Non-linear forecasting as a way of distinguishing chaos from measurement error in time series," Nature, vol. 344, pp. 734–740, 1990.

[25] P. Grassberger and I. Procaccia, "Measuring the strangeness of strange attractors," Physica D: Nonlinear Phenomena, vol. 9, no. 1–2, pp. 189–208, Oct. 1983.

[26] M. E. Cohen, D. L. Hudson, and P. C. Deedwania, "Applying continuous chaotic modeling to cardiac signal analysis," IEEE Eng. Med. Biol. Mag., vol. 15, no. 5, pp. 97–102, Sep./Oct. 1996.

[27] M. Rosenstien, J. J. Colins, and C. J. de Luca, "A practical method for calculating largest Lyapunov exponents from small data sets," Physica D: Nonlinear Phenomena, vol. 65, no. 1–2, pp. 117–134, May 1993.

[28] J. P. Zbilut, N. Thomasson, and C. L. Webber, "Recurrence quantification analysis as a tool for nonlinear exploration of nonstationary cardiac signals," Med. Eng. & Phys., vol. 24, no. 1, pp. 53–60, Jan. 2002.

[29] O. Faust, R. U. Acharya, S. M. Krishnan, and L. C. Min, "Analysis of cardiac signals using spatial filling index and time-frequency domain," BioMed. Eng. OnLine, vol. 3, p. 30, Sep. 2004.

[30] C.-W. Lin, J.-S. Wang, and P.-C. Chung, "Mining Physiological Conditions from Heart Rate Variability Analysis," IEEE Computat. Intell. Mag., vol. 5, no. 1, pp. 50–58, Feb. 2010.

[31] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, et al., "The empirical mode decomposition and Hilbert spectrum for nonlinear and non-stationary time series analysis," in Proc. of the Royal Society A, vol. 454, no. 1971, pp. 903–995, Mar. 1998.

[32] B. Kemp, A. Värri, A. C. Rosa, K. D. Nielsen, and J. Gade, "A simple format for exchange of digitized polygraphic recordings," Electroencephalogr. Clin. Neurophysiol., vol. 82, no. 5, pp. 391–393, May 1992.

[33] S.-F. Liang, H.-C. Wang, and W.-L. Chang, "Combination of EEG Complexity and Spectral Analysis for Epilepsy Diagnosis and Seizure Detection", EURASIP Journal on Advances in Signal Processing, p. 853434, 2010.