International Journal of Engineering

Journal Homepage: www.ije.ir

Chaotic Time Series Recognition: A Deep Learning Model Inspired by Complex Systems Characteristics

A. Pourafzal[a], A. Fereidunian*[a], K. Safarihamid[b]

[a] *Faculty of Electrical Engineering, K. N. Toosi University of Technology, Tehran, Iran*
[b] *School of Electrical Engineering, Iran University of Science and Technology, Tehran, Iran*

PAPER INFO

Paper history:
Received 07 November 2021
Received in revised form 01 October 2022
Accepted 07 October 2022

Keywords:
Chaotic Behavior
Time Series Classification
Deep Learning Models
Fully Convolutional Network
Complex Systems
Lorenz System

ABSTRACT

A deep learning method is developed for chaotic time series classification. We investigate the chaotic state of a dynamical system, based on the output of the system. One of the main obstacles in time series classification is mapping a high-dimensional vector into a scalar value. To reduce the dimensions, it is common to use an average pooling layer block after feature extraction block. This blind process results in models with high computational complexity and potent to overfitting. One alternative is to extract the features manually, then apply shallow learning models to classify the time series. In fact, since complexity lies between the chaos and order, it is a sound idea to refer to complex systems characteristics to explore the chaotic region entrance. Therefore, chaotic state of a dynamical system can be recognized solely based on these characteristics. Inspired by this concept, we conclude that there is a feature space in which the output vector can be sparsified. Thus, we propose a deep learning method which the feature space dimensions successively are reduced in the feature extraction process. Specifically, we employ a fully convolutional network and add on two maximum pooling layers to the relevant feature extraction block. To validate the proposed model, the Lorenz system is employed which exhibits chaotic/non-chaotic states. We generate a labeled dataset containing 10000 samples each with 20000 features of the output of Lorenz system. The proposed model achieves 99.45 percent accuracy over 2000 unseen samples, higher than all the other competitor methods.

doi: *10.5829/ije.2023.36.01a.01*

1. INTRODUCTION

A chaotic state is basically referred to a state of a system, in which no order is observed. In other words, a chaotic system cannot be described with a linear or predictable behavior. However, it complies with the deterministic laws of a nonlinear dynamism. Accordingly, identification of chaotic signals from random ones is a difficult task [1]. In modern science literature, many natural and human-made signals have been recognized as chaotic systems. Hence, the necessity of proficient knowledge on chaos detection becomes quite evident.

Chaos detection in time-series has been of great interest as many real-world problems including human-made or natural systems have chaotic behavior. As some examples of time-series classification applications, rotor analysis in electric machines [2], controlling cardiac chaos [1], load demand forecasting [3], seismographic applications [4], weather forecasting and wind speed prediction [5], forecasting cryptocurrency prices [6] and even the pandemic outbreak [7], can be considered noteworthy.

One of the major challenges in chaotic time series detection is to distinguish between different states of the system. In fact, depending on the parameters of the differential equations, the system experiences different states, having different equilibrium points. Thus, the state of a chaotic system is fully explained by its parameters. Nonetheless, due to the substantial influence of initial values on the output, behavior of the system is unpredictable, and estimation of the parameters solely based on the outputs is rather difficult [8].

To address this issue, several studies have been conducted, which can be categorized into two main

*Corresponding Author Institutional Email: fereidunian@kntu.ac.ir (A. Fereidunian)

Please cite this article as: A. Pourafzal, A. Fereidunian, K. Safarihamid, Chaotic Time Series Recognition: A Deep Learning Model Inspired by Complex Systems Characteristics, ***International Journal of Engineering, Transactions A: Basics***, Vol. 36, No. 01, (2023), 1-9

groups, namely, model-based approaches and data-driven approaches.

Within model-based approaches, Barahona and Poon [9] have suggested a method based on the prediction feature. Later, Kodba et al. [10] presented Lyapunov exponent was as a powerful feature. Thereafter, Wernecke et al. [11] suggested a modification of this method with higher performance. However, the error and computational complexity of all these model-based methods depend vastly on how well the chosen model has been fitted.

Data-driven approaches are more comprehensible compared to model-based approaches; we believed. Gottwald and Melbourne [12] have introduced first the correlation test which performs well for noise-free scenarios and seeks for chaotic behavior in a given deterministic dynamical system. Then the regression test was introduced to reduce the noise. Tempelman and Khasawneh [13] have recommended a framework in which both correlation and regression tests were utilized. Also, Bhattacharya and Ray [14] utilized Hidden Markov Models (HMM) classification to identify chaos within the time-series of different attractors. Khosravi and Gholipour [15] estimated the Lorenz system parameters through metaheuristic algorithms. Kirichenko et al. [16] utilized the Hurst exponent as a feature to be fed into Random Forest and neural network methods to detect fractals.

Moreover, approaches by Pourafzal and Fereidunian [17] and Safarihamid et al. [18] took advantages of the link between complex systems and chaotic systems. According to Poincaré, the unpredictable behaviour of non-linear dynamical systems can be interpreted as an extreme point of complexity instead of disorder [19]. Thus, in this group of approaches, complex system features are utilized to detect chaotic time series. Pourafzal and Fereidunian [17] stated the most important features of complex systems as emergence, self-organization, predictability, and complexity. Then, information theoretic definitions of such characteristics were investigated and employed for chaos recognition.

Additionally, Safarihamid et al. [18] reviewed the definitions of characteristics and they substituted with measured-theoretic entropies (based on Kolmogorov complexity). Specifically, it is shown that emergence is indeed the violation of regularities in a time-series, whereas self-organization can be defined as the expectation of limited outcomes in a system. Then, they utilized these properties to develop a joint-entropy time series classifier with better performance compared to single entropy classifiers.

Deep learning models are considered as another member of data-driven approaches. These models provide an end-to-end framework which avoids a lot of heavy pre-processing and feature extractions [20]. Architectures like Long Short-Term Memory (LSTM) network [21], Deep multilayer perceptron (MLP) [22], Residual Networks (ResNet) [22], Multi Scale Convolutional Neural Networks (MCNN) [23] and Fully Convolutional Networks (FCN) [22] are among the most important models which could achieve desired performance without any prior feature extraction.

MLP constitutes one of the most common architectures, where all the neurons in the current layer are fully connected to the former one. The main issue with such networks is that each neuron will be associated with a specific time stamp in the time series and thus the temporal correlations between datapoints would be ignored [22]. CNNs are usually employed in 2D classifications, such as image classification applications. However, this structure could act as a moving average filter in one dimensional domain, following by a proper activation function [22]. The architecture of ResNet is the deepest one (with 11 layers), among all these candidates. It leverages from a residual block as a shortcut, which avoids vanishing gradient by connecting the shallow layers to the output [22]. Theoretically, MCNN could reach to the best performance among all the models [24]. However, the large extraction of hyper-parameters in such models makes it almost impossible to find the best solution.

1. 1. Motivation We realize that the number of trainable parameters in deep learning models (number of filters, kernels, and depth of the network) are chosen based on a trial-and-error process. However, as suggested by Pourafzal and Fereidunian [17] and Safarihamid et al. [18], regardless of observation length of chaotic time series, they can be classified using a few features of the complex system [17, 18]. This gives us the intuition that there is a feature space in which the given chaotic time series can be sparsified. Thus, we employed the Fully Convolutional Networks (FCN) proposed by Boullé et al. [25] and utilize two Max Pooling layers in the feature extraction block. By this mean, we prune all the redundant insignificant features in the feature extraction block, resulting in reducing the signal dimensions. The proposed deep model can be observed as the deep counterpart of the proposed shallow learning model by Pourafzal and Fereidunian [17]. Supported by the simulations, it is shown that the proposed deep model can reach to 99.45 percent accuracy over unseen test data, higher than other state-of-the-art methods compared in the text.

The rest of this paper is organized as follows. In Section 2, chaos recognition with complex system theory will be explained and some of the most important features of a complex system will be mentioned. In section 3, the proposed deep learning model is developed. In section 4, implementation of each method as well as the time series generation will be explored. In section 5, the simulation results and performance of the proposed method in

comparison with other methods is stated. We conclude the paper in section 6.

2. CHAOS RECOGNITION USING COMPLEX SYSTEMS CHARACTERISTICS

In this section, we briefly discuss the intuition behind the proposed deep learning model. In the following, we discussed the utilized model as well as the features used to detect chaos in a given time series.

2. 1. Lorenz System We employed Lorenz system as a well-known chaotic system, to generate a chaotic time-series. The Lorenz system is defined in three-dimensional space by three parameters of ρ, σ and β [26]. The parameter ρ which is proportional to Rayleigh number [26], can specify the state of the system alone. Based on Hopf bifurcation, we have the following condition [27]

$$\rho < \sigma \frac{\sigma+\beta+3}{\sigma-\beta-1}. \quad (1)$$

With violation of the Hopf bifurcation condition, the system loses the stability of equilibrium point which results in chaotic behavior [26].

2. 2. Feature Extraction In data-driven approaches, each state is classified based on the time-series itself, rather than estimating the parameters. On average these approaches result in better performance compared to model-based approaches. However, the dimensions of the feature space are the identical to the input time series, which could be a burden on the time and space capacity of the system.

To reduce these complexities, proper feature-space reduction preprocessing units are beneficial. In feature engineering, having a prior knowledge about the number of features helps us avoiding the overfitting problem. Also, a model with less features to train is much lower time and space complexity.

Motivated by Pourafzal and Fereidunian [17], we investigate the dimensions of features in a chaotic time series employing complex system features. For a complex system, different number of features are mentioned in the literature. For instance, Grus et al. [28] have studied up to 13 different attributes related to complex systems. Here, the same as Pourafzal and Fereidunian [17], we investigate Emergence, Complexity, Predictability, Self-organization and Sensitivity to initial conditions [8] as the most cited features of a complex system.

2. 2. 1. Unpredictability As one of the main characteristics of complex systems, unpredictability is defined as being incapable of calculating the future samples of the system according to the past observations, without precise information about its initial conditions.

Different methods have been suggested to measure unpredictability. Diebold and Kilian [29] proposed a measure based on the precision of forecasting future samples. Another well-known measure is the Hurst exponent, which is fully discussed by Hurst [30], and takes a value in the range of [0,1]. This value categorizes the system based on its behavior, into three different states, indicating the long-term tendency of the signal to follow its current trends.

2. 2. 2. Emergence According to the systems theory, the components of a system which do not have a certain attribute, may show that behavior when working as a whole system. For instance, the color of an element is cannot be observed in the atoms of that element [31]. In the information theoretical literature, emergence can be seen as the difference between the output and input information of the system, which are related to the state of the system and its initial conditions, respectively. This means that the additional produced information has been emerged within the system as a result of the interaction between its inner subsystems. Bearing this in mind, it can be concluded that different entropy levels, could be interpreted as a measure of emergence within the system. Accordingly, the emergence was defined by Gershenson and Fernández [32] stated as follows:

$$E = \frac{H_{output}}{H_{input}}$$
$$H \triangleq -\sum p(X) \log p(X) \quad (2)$$

2. 2. 3. Self-organization A self-organized system is able to find its way through forming a specific structure or pattern without any external effects and only due to the nature of its components [8]. In other words, once a pattern is observed within the system, the internal particles of the system will find their way to form that pattern again, regardless of the external situations of the system. Therefore, linearity in a system can be seen as a completely self-organizing behavior. According to this, it could be claimed that self-organization is inversely proportional to the entropy, as the information production rate.

2. 2. 4. Complexity The above-mentioned features are not capable of identifying chaotic behavior, as emergence fails in the presence of noise, and self-organization identifies linear systems as the most self-organized systems. Hence, Gershenson and Fernández [32] have suggested to define complexity as a combination of these two features, by multiplying them. Alternatively stated, the maximum complexity occurs between the extremes of emergence and self-organization.

2. 3. Shalow Learning Classification To assess the strength of these features, we applied them as inputs to four machine learning models, namely support vector machine (SVM), K-Nearest Neighbors (KNN), Multilayer Perceptron (MLP) and Random Forest. Then, we measured the accuracy of classification on a test data.

3. PROPOSED DEEP LEARNING MODEL

The main advantage of deep learning methods in time-series classification is that we can feed the raw data without any preprocessing into the machine. This is due to its automatic feature extraction. Nevertheless, aiding the machine to select a subset of features can improve the time complexity and avoid overfitting.

The overall architecture of this network is provided in Figure 1. This network consists of three blocks of feature extraction, pooling, and classification. We assumed that most of the features extracted in the feature extraction block are redundant, and the feature space dimensions could be reduced. Thus, at the feature extraction block, we employed Max Pooling layers between the convolutional layers (CL) to prune the features at each layer.

3. 1. Feature Extraction Block In this layer, firstly, three 1-D CL are utilized, each with 10 feature maps, kernel size of three, and stride of one, to learn the feature space, followed by a Batch Normalization (BN) unit to speed up the convergence. Also, to avoid overfitting problem by selecting specific neurons to update at each iteration, Rectified Linear Unit (ReLU) as the activation function is chosen. Then, between the CLs, we take advantage of a Max Pooling layer, which reduces the dimensions step by step. Assuming the input is vector $\mathbf{x}_0$, the feature extraction block is described as follows [33]:

$$\begin{aligned} &\mathbf{y}_i = \mathbf{h}_i * \mathbf{x}_{i-1} + \mathbf{b}_i \\ &\mathbf{o}_i = \sigma\big(f(\mathbf{y}_i)\big) \\ &\mathbf{x}_i = \mathcal{M}(\mathbf{o}_i, K_i) \\ &\forall i \in \{1,2\} \end{aligned} \tag{3}$$

where $\mathbf{h}_i$ is the i-th convolutional layer, $\mathbf{b}_i$ is bias term, $f(.)$ is the batch normalization [34], $\sigma(.)$ is activation layer and $\mathcal{M}(., K_i)$ is the max pooling layer with size of K_i. This procedure assists the model to learn the overall pattern better by dismissing the unimportant features. Although using this architecture the learning curve increases slower than the conventional FCN method, it expresses faster performance in evaluation phase.

3. 2. Average Pooling Block Subsequently, the outputs of the feature extraction block are fed into a global average pooling layer prior to the classifier. Pooling layers create a response for each feature map, which reduce the dimensions without overfitting problem of the flattening layer. Traditionally, the global average pooling layer penalize the feature extraction block to dismiss the output features with insignificant values. However, the error fails to propagate properly when the rank of feature matrix is much higher than the pooling layer output.

Here, by employing two Max pooling layer in the feature extraction block, we ensure that the feature matrix is sparse. Thus, there is no need for skip connection.

3. 3. Classification Block Finally, in the last block, three fully connected layers with the size of 10, 5 and 2, in addition to a sigmoid activation layer are employed. The output of this layer is evaluated for binary classification.

4. IMPLEMENTATION AND RESULTS

4. 1. Dataset Generation To train and test the machine learning models, a dataset based on the Lorenz attractor is collected. Specifically, we perform 10000 independent simulation trial with the following specifications. We set the parameters $\sigma = 10$ and $\beta = 8/3$, and the observation time window is 20 seconds for each trial. The parameter ρ is selected with uniform distribution from the set of $\{0.1, 0.2, \ldots 44.9\}$. Moreover, the initial conditions of each trial are standard white Gaussian random vectors $N(0,1)$ in 3-D space. Finally,

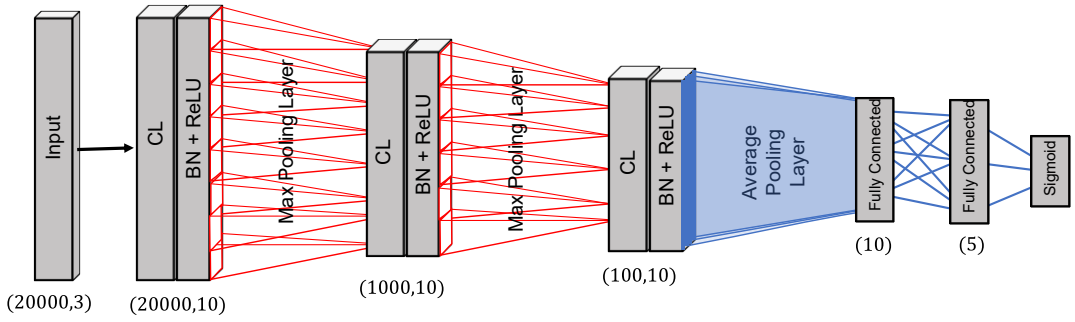

Figure 1. Deep Learning Network Architecture

the numerical method applied on the ordinary differential equations has the step-size of 10^{-3}s.

Chaotic behavior is among the three different phases that Lorenz attractor experiences in which the value ρ violets the inequality of Equation (1). Likewise, we selected Hopf condition as an ideal candidate for the Target of our binary classification.

$$\begin{matrix} Target = 1 \\ \rho \gtrless 24.7 \\ Target = 0 \end{matrix} \quad (4)$$

Thus, the final dataset consists of two data frames, an input with the size of $(10000, 20000, 3)$ and the targets with the size of $(10000,)$, respectively.

The computer for the simulation is a Core-i7 @ 2.70 GHz with 16 GB of RAM for the tasks on CPU setting in addition to 12 GB of GPU on Tesla K80 for deep learning. The dataset is generated and collected in a comma-separated values (CSV) file, with the volume of 5 Giga Byte (GB).

4. 2. Complex Systems Approach In the complex system approach discussed in section 2, the features are extracted from the collected dataset and then, they are fed into the shallow learning models.

We extract the predictability with Hurst exponent [17] with 20 lags, self-organization with Disequilibrium [17] with $K = 10$ and emergence using distribution entropy with 10 frequency bins [18]. In addition, complexity is the product of emergence and self-organization.

The candidates for shallow learning classifiers are as follows. In SVM models, the applied hyper parameters are linear, polynomial, and radial basis function (RBF), $C = \{1, 10, 100, 200\}$ for different penalty parameters, and the value of Gamma is chosen within the set of $\{10^{-3}, 10^{-4}\}$. For the KNN classifier, the numbers of neighbours are selected within the range of $\{3, 4, \dots, 30\}$. as the distance metric of this classifier. We also utilized MLP with Adam solver as a posterior to our work [35]. The hyper-parameters in this model are chosen as 50 and 100 as different numbers of hidden layers, {sigmoid, ReLU} as activation functions, and $1 \times 10^{-3}$and 5×10^{-2} as different regularization terms (L2 penalty). We also adopted different learning rate schedules of constant and adaptive. In Random Forest, different combinations of decision trees are utilized to find the highest performance. The investigated hyper parameters are the total number of internal classifiers in the set of $\{100, 150, 200, 250\}$ and maximum depth of each tree in the set of $\{5, 10, 15, 20, 30\}$.

The dataset is shuffled and separated into train/test subsets with proportion of 80%, 20%, respectively.

The classification report using shallow learning methods, as well as the details about the best hyper-parameters, are depicted in Table 1. As it is apparent the shallow learning models provide good performance in either precision or recall. This is due to the loss of information between feature extraction and classification. In fact, as we calculate the features manually, accumulation over all the available samples without proper weighting factor will cause a uniform importance which cause destroying the information.

The consumed time in a simulation run is averaged over 5000 time series. Each Hurst exponent requires 1.43×10^{-3}seconds to perform, as well as 2.11×10^{-2} for emergence and 4.7×10^{-4} for the self-organization measure. In training phase, on average, SVM takes 5.5×10^{-1} [s], KNN requires 2.3×10^{-3}, 9.8×10^{-1} for Random Forest as well as 1.6×10^{-1} for MLP. Also in prediction phase, average times are $2.5 \times 10^{-4}, 5.7 \times 10^{-4}, 6 \times 10^{-3}$ and 9.06×10^{-5}, respectively.

4. 3. Feature Importance In this section, we obtain the importance of extracted features utilizing the Mutual Information (MI) [36] between features well as the Mean Decrease in Impurity (MDI) of Random Forest classifier.

Specifically, we calculate the two metrics (MI and MDI) for the features of predictability (Hurst), self-organization (Disequilibrium), emergence (Entropy), complexity and initial condition for each axis (x, y, z) of the dataset.

The bar plots of feature importance are given in Figure 2. Overall, all features except for initial condition seem to be crucial for the classification as none of them were declined in these results. However, emergence stands at higher level in both metrics, followed by predictability and Self-organization, just about 25 percent each. Moreover, x-axis and y-axis contain more information compared to z-axis. This is important for low-complexity applications due to high-volume of each axis in computation and memory allocation. It is worth mentioning that emergence and self-organization are among the most important characteristics of complex systems supporting the idea of using these characteristics in chaos recognition [17].

4. 4. Proposed Deep Learning Model Implementation Here instead of train/test split, we split the shuffled dataset into three subsets of train (60%), validation (20%) and test (20%). The dataset contains 60000 features to classify the data, while there are only 10000 samples of different time-series available. This well-known data-dimensionality problem as it is mentioned in section 3.

We train the proposed deep model using the train set for 100 epochs. For validating the model at each epoch, we use the validation set. For both, the batch size is set to 750 with Adam optimizer introduced by Kingma and Ba [37].

TABLE 1. Performance of Conventional Machine Learning methods + Feature Extraction

Model	Selected Hyper Parameters	Performance				
		Accuracy	Precision		Recall	
			0	1	0	1
SVM	$C = 100$ $Kernel = Linear$ $\gamma = 10^{-3}$	0.75	0.97	0.54	0.16	0.99
KNN	27 $Neighbors$	0.77	0.55	0.98	1	0.18
MLP	$Activation: ReLU,$ $layers = 100, alpha = 10^{-3}$ $learning\ rate = adaptive$	0.74	0.95	0.54	0.16	0.99
Random Forest	$Max\ depth = 30$ $Estimators = 200$	0.75	0.96	0.54	0.16	0.99

We tune the model using two hyper parameters of K_1 and K_2, the pool size of first and second max pooling layer in the feature extraction block, respectively. In addition, we optimal value of learning rate is explored in 100 points linearly spaced the range of $(10^{-4}, 1)$.

Dimensions of the output of feature extraction block as well as the final performance of model associated with each pool size are shown as pairs, respectively, in Table 2. It can be observed that by reducing the dimensions from 10000 to 50 the performance degrades up to 10 percent accuracy, which supports the redundance of input features. In addition, the performance degrades more when we reduce the dimensions in later stages. Indeed, employing only one global averaging (as used by Boullé et al. [25]) at the output of feature extraction block can be interpreted as the worst-case scenario of this experiment. Based on this table, we choose $K_1 = 20, K_2 = 10$ to set a trade-off between performance and dimensions.

Figure 3 illustrates the dependency of performance on the learning rate for the selected pool sizes. Based on this figure, we select 2×10^{-3} as the optimal value of learning rate.

To evaluate the time-complexity improvement when the Max pooling layer is employed, we compare the consumed time of proposed deep model against FCN proposed by Boullé et al. [25]. For both deep learning models, there are an overall of 957 trainable parameters.

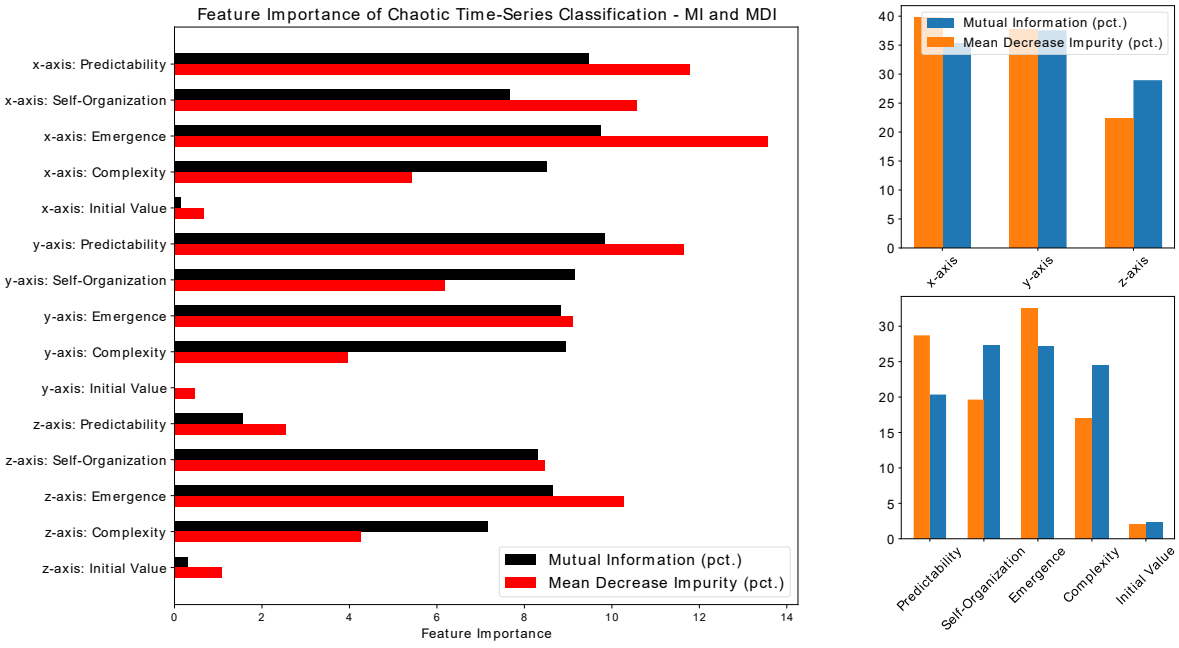

Figure 2. Feature importance of extracted complex system features

TABLE 2. Parameter selection of Max Pooling Layer for the proposed method

	$K_2 = 2$	$K_2 = 4$	$K_2 = 10$	$K_2 = 20$
$K_1 = 2$	(5000, 0.79)	(2500, 0.75)	(1000, 0.77)	(500, 0.78)
$K_1 = 4$	(2500, 0.79)	(1250, 0.78)	(500, 0.78)	(250, 0.74)
$K_1 = 10$	(1000, 0.78)	(500, 0.8)	(200, 0.8)	(100, 0.74)
$K_1 = 20$	(500, 0.79)	(250, 0.78)	(100, 0.78)	(50, 0.72)

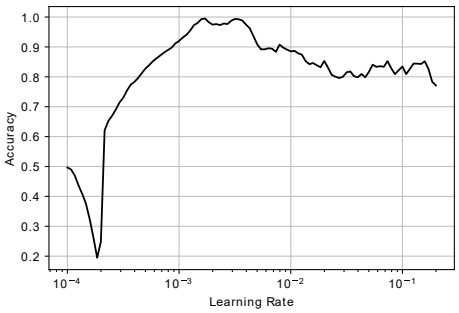

Figure 3. Dependency of Accuracy on Learning rate

By enabling the GPU in the simulations, approximately $7 \times 10^{-3}[s]$ is required for each batch in the training phase given by Boullé et al. [25]. While this value is $6 \times 10^{-3}[s]$ for the proposed model. In addition, the classification could be performed by $3 \times 10^{-3}[s]$ in evaluation phase. To make a fair comparison with shallow learning methods, we also provide the consumed time when only CPU is available. In these setting, each batch of data in training model by Boullé et al. [25] requires $1.02 \times 10^{-1}[s]$, as well as 19 $[s]$ for an epoch. On the contrary, when we utilize Max Pooling, these values are reduced to $4 \times 10^{-2}[s]$ and 8 $[s]$, respectively. Moreover, in the evaluation of these models in the same setting, $6.2 \times 10^{-3}[s]$ is required for the average of a classification task using FCN [25], while it requires $5.2 \times 10^{-3}[s]$ for the proposed deep model.

Finally, we compared our proposed deep learning model with FCN proposed by Boullé et al. [25], the FCN network proposed by Wang et al. [22] and deep MLP also by Wang et al. [22]. For fair comparison with Boullé et al. [25], we select the same parameters as the proposed network, while the only difference is max pooling layer. For Wang et al. [22], we employ three CL layers with $\{32,64,32\}$ filters each with $\{8,5,3\}$ kernels. Also, for the MLP we choose three fully connected layer with the size of $\{200,200,1\}$ and two dropout layers with fractions of $\{0.1, 0.1\}$.

In Figure 4, the learning curves of different models are illustrated. As expected, the proposed model (red solid line) requires more epochs to learn the data. However, it converges to a value higher than other deep network.

Finally, to evaluate the performance of proposed deep network, we give the 2000 test observation as the input to all the trained networks and validate the output labels against the actual ones. In Table 3, number of correct labels as well as the accuracy are reported. The proposed network outperforms other networks with 1989 correct labels, which indicates the strength of model in classification.

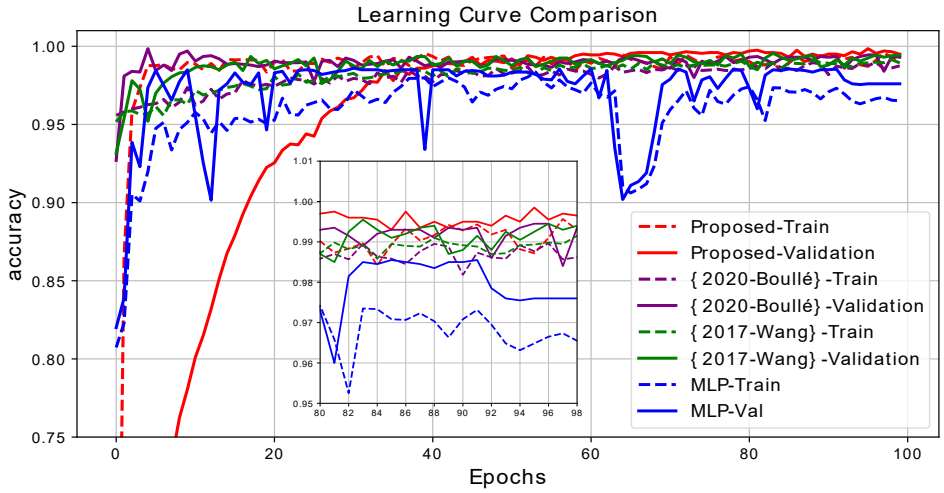

Figure 4. Comparison between the learning curves of Different Deep Learning models

TABLE 3. Performance of Deep Models on Unseen Test Data

	Number of correct labels (from 2000)	Accuracy
Proposed	1989	0.9945
FCN [25]	1984	0.992
FCN [22]	1987	0.9935
MLP [22]	1945	0.9725

5. CONCLUSIONS

Recognition of chaotic time series plays an important role in engineering and science as a lot of real-world signals (e.g., earthquakes, biomedical signals, and climatic data) could be modeled within this certain group. Complex systems as one of the well-known essentials in systems theory, share some considerable similarities with the chaotic models. Specifically, a lot of chaotic events can be explained based on only four features of a complex system, namely, prediction, emergence, self-organization, and complexity.

This similarity inspired us to seek the characteristic of a complex system in a chaotic time series. Therefore, we developed a network in which instead of blind feature extractions, we gradually reduce the dimensions of features. This is in fact similar to manual feature extraction done by complex system-based time series classifications yet is more efficient in computational complexity and accuracy. The simulation results indicate an overall performance of 99.45 percent accuracy for the proposed deep network, which is higher compared to all the other methods discussed in this paper.

6. REFERENCES

1. Lorenz, E.N. and Haman, K., "The essence of chaos", ***Pure and Applied Geophysics***, Vol. 147, No. 3, (1996), 598-599.
2. Modaresahmadi, S., Ghazavi, M. and Sheikhzad Saravani, M., "Dynamic analysis of a rotor supported on ball bearings with waviness and centralizing springs and squeeze film dampers", ***International Journal of Engineering, Transactions C: Aspects***, Vol. 28, No. 9, (2015), 1351-1358. doi: 10.5829/idosi.ije.2015.28.09c.13
3. Al-ani, B. and Erkan, E., "A study of load demand forecasting models in electricity using artificial neural networks and fuzzy logic model", ***International Journal of Engineering, Transactions C: Aspects***, Vol. 35, No. 6, (2022), 1111-1118. doi: 10.5829/ije.2022.35.06c.02
4. Sparrow, C., "The lorenz equations: Bifurcations, chaos, and strange attractors, Springer Science & Business Media, Vol. 41, (2012). doi: 10.1002/zamm.19840640122
5. Chinforoush, N. and Latif Shabgahi, G., "A novel method for forecasting surface wind speed using wind-direction based on hierarchical markov model", ***International Journal of Engineering, Transactions B: Applications***, Vol. 34, No. 2, (2021), 414-426. doi: 10.5829/ije.2021.34.02b.13
6. Derbentsev, V., Babenko, V., Khrustalev, K., Obruch, H. and Khrustalova, S., "Comparative performance of machine learning ensemble algorithms for forecasting cryptocurrency prices", ***International Journal of Engineering, Transactions A: Basics***, Vol. 34, No. 1, (2021), 140-148. doi: 10.5829/ije.2021.34.01a.16
7. Babaei, A., Jafari, H., Banihashemi, S. and Ahmadi, M., "Mathematical analysis of a stochastic model for spread of coronavirus", ***Chaos, Solitons & Fractals***, Vol. 145, (2021), 110788. doi: 10.1016/j.chaos.2021.110788
8. Fereidunian, A., Lesani, H., Zamani, M.A., Kolarijani, M.A.S., Hassanpour, N. and Mansouri, S.S., "A complex adaptive system of systems approach to human–automation interaction in smart grid", ***Contemporary Issues in Systems Science and Engineering***, (2015), 425-500. doi: 10.1002/9781119036821.ch12
9. Barahona, M. and Poon, C.-S., "Detection of nonlinear dynamics in short, noisy time series", ***Nature***, Vol. 381, No. 6579, (1996), 215-217. doi: 10.1038/381215a0
10. Kodba, S., Perc, M. and Marhl, M., "Detecting chaos from a time series", ***European Journal of Physics***, Vol. 26, No. 1, (2004), 205. doi: 10.1088/0143-0807/26/1/021
11. Wernecke, H., Sándor, B. and Gros, C., "How to test for partially predictable chaos", ***Scientific reports***, Vol. 7, No. 1, (2017), 1-12. doi: 10.1038/s41598-017-01083-x
12. Gottwald, G.A. and Melbourne, I., "On the implementation of the 0–1 test for chaos", ***SIAM Journal on Applied Dynamical Systems***, Vol. 8, No. 1, (2009), 129-145. doi: 10.1137/080718851
13. Tempelman, J.R. and Khasawneh, F.A., "A look into chaos detection through topological data analysis", ***Physica D: Nonlinear Phenomena***, Vol. 406, (2020), 132446. doi: 10.1016/j.physd.2020.132446
14. Bhattacharya, C. and Ray, A., "Data-driven detection and classification of regimes in chaotic systems via hidden markov modeling", ***ASME Letters in Dynamic Systems and Control***, Vol. 1, No. 2, (2021). doi: 10.1115/1.4047817
15. Khosravi, A. and Gholipour, R., "Parameter estimation of loranz chaotic dynamic system using bees algorithm", ***International Journal of Engineering, Transactions C: Aspects***, Vol. 26, No. 3, (2013), 257-262. doi: 10.5829/idosi.ije.2013.26.03c.05
16. Kirichenko, L., Radivilova, T. and Bulakh, V., "Binary classification of fractal time series by machine learning methods", in International Scientific Conference "Intellectual Systems of Decision Making and Problem of Computational Intelligence", Springer. (2019), 701-711. doi: 10.1007/978-3-030-26474-1_49
17. Pourafzal, A. and Fereidunian, A., "A complex systems approach to feature extraction for chaotic behavior recognition", in 2020 6th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS), IEEE. (2020), 1-6. doi: 10.1109/ICSPIS51611.2020.9349551
18. Safarihamid, K., Pourafzal, A. and Fereidunian, A., "A joint-entropy approach to time-series classification", in 2021 7th International Conference on Signal Processing and Intelligent Systems (ICSPIS), IEEE. (2021), 1-7. doi: 10.1109/ICSPIS54653.2021.9729371
19. Hayles, N.K., "Chaos bound: Orderly disorder in contemporary literature and science, Cornell University Press, (2018).
20. Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L. and Muller, P.-A., "Deep learning for time series classification: A review", ***Data Mining and Knowledge Discovery***, Vol. 33, No. 4, (2019), 917-963. doi: 10.1007/s10618-019-00619-1
21. Zhou, T., Chu, C., Xu, C., Liu, W. and Yu, H., "Detecting predictable segments of chaotic financial time series via neural

network", ***Electronics***, Vol. 9, No. 5, (2020), 823. doi: 10.3390/electronics9050823

22. Wang, Z., Yan, W. and Oates, T., "Time series classification from scratch with deep neural networks: A strong baseline", in 2017 International joint conference on neural networks (IJCNN), IEEE. (2017), 1578-1585. doi: 10.1109/IJCNN.2017.7966039
23. Pei, S.-C. and Tseng, C.-C., "Complex adaptive iir notch filter algorithm and its applications", ***IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing***, Vol. 41, No. 2, (1994), 158-163. doi: 10.1109/82.281849
24. Cui, Z., Chen, W. and Chen, Y., "Multi-scale convolutional neural networks for time series classification", arXiv preprint arXiv:1603.06995, (2016). doi: 10.48550/arXiv.1603.06995
25. Boullé, N., Dallas, V., Nakatsukasa, Y. and Samaddar, D., "Classification of chaotic time series with deep learning", ***Physica D: Nonlinear Phenomena***, Vol. 403, (2020), 132261. doi: 10.1016/j.physd.2019.132261
26. Lorenz, E.N., "Simplified dynamic equations applied to the rotating-basin experiments", ***Journal of the Atmospheric Sciences***, Vol. 19, No. 1, (1962), 39-51. doi: 10.1175/1520-0469(1962)019<0039:SDEATT>2.0.CO;2
27. Marsden, J.E. and McCracken, M., "The hopf bifurcation and its applications, Springer Science & Business Media, Vol. 19, (2012). doi: 10.1007/978-1-4612-6374-6
28. Grus, L., Crompvoets, J. and Bregt, A.K., "Spatial data infrastructures as complex adaptive systems", ***International Journal of Geographical Information Science***, Vol. 24, No. 3, (2010), 439-463. doi: 10.1080/13658810802687319
29. Diebold, F.X. and Kilian, L., "Measuring predictability: Theory and macroeconomic applications", ***Journal of Applied Econometrics***, Vol. 16, No. 6, (2001), 657-669. doi: 10.1002/jae.619
30. Hurst, H.E., "Long-term storage capacity of reservoirs", ***Transactions of the American Society of Civil Engineers***, Vol. 116, No. 1, (1951), 770-799. doi: 10.1061/TACEAT.0006518
31. Anderson, P.W., "More is different: Broken symmetry and the nature of the hierarchical structure of science", ***Science***, Vol. 177, No. 4047, (1972), 393-396. doi: 10.1126/science.177.4047.393
32. Gershenson, C. and Fernández, N., "Complexity and information: Measuring emergence, self-organization, and homeostasis at multiple scales", ***Complexity***, Vol. 18, No. 2, (2012), 29-44. doi: 10.1002/cplx.21424
33. Chollet, F., "Deep learning with python, Simon and Schuster, (2021).
34. Ioffe, S. and Szegedy, C., "Batch normalization: Accelerating deep network training by reducing internal covariate shift", in International conference on machine learning, PMLR., (2015), 448-456.
35. Lucas, C., Lesani, H. and Fereidunian, A., "Distribution systems reconfiguration using pattern recognizer neural networks", ***International Journal of Engineering***, Vol. 15, No. 2, (2002), 135-144.
36. Torkkola, K., "Feature extraction by non-parametric mutual information maximization", ***Journal of Machine Learning Research***, Vol. 3, No. Mar, (2003), 1415-1438.
37. Kingma, D.P. and Ba, J., "Adam: A method for stochastic optimization", arXiv preprint arXiv:1412.6980, (2014). doi: 10.48550/arXiv.1412.6980

Persian Abstract

چکیده

آشکارسازی سیگنال‌های آشوبی با توجه به کاربرد آن‌ها در علوم مختلف یکی از مسائل مورد توجه در دهه های اخیر بوده است. امروزه با گسترش فناوری، سیستم‌های مبتنی بر یادگیری ماشین قادرند تا این سیگنال‌ها را با دقت بالایی دسته بندی کنند. در این مقاله، یک روش مبتنی بر یادگیری عمیق به منظور طبقه بندی این سیگنال‌ها ارائه شده است. طبقه بندی سری های زمانی تنها با بهره گیری از شبکه‌های عمیق و بدون توجه به ساختار تنک داده موجب بالا رفتن بار محاسباتی سیستم می‌شود. از این رو به منظور کاهش بار محاسباتی و با توجه به ارتباط تنگاتنگ آشوب و سیستم‌های پیچیده به این موضوع پی بردیم که می‌توان ورودی را در یک فضای تنک ترسیم کرد. ما یک شبکه عمیق جدید ارائه کردیم که در بلوک استخراج ویژگی آن، ابعاد ماتریس ورودی را با کمک لایه‌ی Max Pooling در چند مرحله کاهش می‌دهیم. سپس با تولید یک سری زمانی از سیستم لورنز، روش ارائه شده را به بوته آزمایش گذاردیم. شبیه سازی‌ها نشان می‌دهد روش ما برتر از دیگر روش‌ها قابلیت طبقه بندی سری‌های زمانی را تا دقت ۹۹.۴۵ درصد داراست.