Identifying Chaotic Dynamics in Noisy Time Series through Multimodal Deep Neural Networks

Alessandro Giuseppi[1]‡, **Danilo Menegatti**[1], **Antonio Pietrabissa**[1]

[1] Department of Computer, Control and Management Engineering of the University of Rome La Sapienza, Via Ariosto 25, 00185, Rome, Italy.

E-mail: giuseppi@diag.uniroma1.it

April 2024

Abstract. Chaos detection is the problem of identifying whether a series of measurements is being sampled from an underlying set of chaotic dynamics. The unavoidable presence of measurement noise significantly affects the performance of chaos detectors, as discerning chaotic dynamics from stochastic signals becomes more challenging. This paper presents a computationally efficient multimodal deep neural network tailored for chaos detection by combining information coming from the analysis of time series, recurrence plots and spectrograms. The proposed approach is the first one suitable for multi-class classification of chaotic systems while being robust with respect to measurement noise, and is validated on a dataset of 15 different chaotic and non-chaotic dynamics subject to white, pink or brown coloured noise.

Chaos Detection, Chaotic dynamics, Deep Neural Networks
Submitted to: *Meas. Sci. Technol.*

1. Introduction

Chaos is a behaviour that affects several deterministic nonlinear dynamical systems which exhibit complex, random-looking, evolutions due to a significant sensitivity to small changes in their initial conditions.

When studying a time series of observations, the problem of *chaos detection* consists in understanding whether the underlying dynamics is of a chaotic nature or not, and it is crucial for the correct analysis of the system and to predict its future evolution. In fact, discerning deterministic, albeit chaotic, dynamics from stochastic processes opens up several opportunities in fields that can benefit from having a reliable prediction regarding the future evolution of a complex system, such as is the case for financial market analysis, weather forecast, epidemiology and neurosciences [1] with significant impacts on entire industrial fields and markets. Detecting chaos in empirical data is a

‡ corresponding author

particularly complex task, as the unavoidable presence of noise affecting the observations may render the discernment of chaotic dynamics from periodic deterministic trajectories and stochastic signals particularly challenging [2].

In this direction, this paper presents an ad-hoc multimodal deep neural network (DNN) architecture tailored for detecting and classifying chaotic dynamics in noisy time series. The proposed DNN employs standard tools for chaos analysis, namely recurrence plots and spectrograms, to recognize the underlying dynamics from partial state observations corrupted by stochastic noise. The proposed DNN architecture is then validated on a dataset derived from the one originally presented in [3] to discern among 15 different dynamics subject to various levels and types of stochastic noise.

This study improves existing literature in several aspects:

- The proposed DNN treats chaos detection as a multi-class classification problem, whereas most of the machine learning-based approaches in the literature only focus on a single set of chaotic dynamics following a binary approach.
- The developed system shows a high level of tolerance against measurement noise.
- The DNN allows for computationally efficient chaos detection, as the computing burden is limited to its off-line training phase.

The remainder of the paper is organized as follows: Section 2 presents a survey of the relevant literature; Section 3 introduces some needed preliminaries and definitions; Section 4 describes and explains the proposed neural network architecture; Section 5 discusses the results of the validation of the designed DNN carried out on a recent public dataset; Section 6 draws the conclusions and highlights possible future works.

Notation: $\mathbf{A}_{i,j}$ denotes the j-th element of the i-th row of the matrix $\mathbf{A}$; the operator $||\cdot||$ denotes the L^2-norm.

2. Related Works

Chaos theory is the branch of systems theory that deals with the analysis and study of chaotic dynamical systems and their trajectories. A chaotic system is characterized by deterministic dynamics that amplify exponentially small perturbations of its initial conditions, meaning that small differences in the system's initial state lead to significantly different trajectories causing what is commonly known as the *butterfly effect* [2].

Over the years, due to the ubiquitous presence of chaos in natural processes, chaos theory has caused a significant impact in almost all fields of engineering, spacing from fluid and plasma dynamics [4] to neuroscience [5], optics [6] and medicine [7]. Chaos detection is the problem of understanding whether or not a set of observations is sampled from a dynamical system that is characterized by chaotic deterministic dynamics, allowing for proper system analysis and potentially the design of adequate control laws.

Assuming *a priori* knowledge on the structure of the system dynamics, a first approach for chaos detection involves the determination of which regions of the system's

parameter space lead to chaotic behaviours. In this setting, a common approach to determine such regions is based on the analytical determination of the maximal Lyapunov Characteristic Exponent (LCE) [8] to estimate how fastly neighbouring trajectories diverge with time.

A different approach to chaos detection can be followed in a data-driven setting when no information is available on the system dynamics, and one is interested in determining whether a given time series of observations has been produced by a chaotic system. In such a scenario, it is still possible to estimate the LCE directly from the time series [8,9], but the estimation sensitivity to measurement noise and its limitations in terms of sample and computational requirements has led scientists to design more robust and efficient solutions [2, 10]. In this direction, over the last few years several works investigated how machine and deep learning solutions [3,11–16] may contribute to the chaos detection problem. The authors in [11–13] tested various DNN architectures as binary classifiers fed directly with the observed time series to detect various different chaotic dynamics. In [11, 12] it was assumed to have *a priori* knowledge on the system dynamics, as each DNN was specialized for a single system, while [13] tests how a DNN trained for binary chaos detection of the Chirikov standard map performs in recognizing chaos also on the Logistic map and Lorenz system. In [3] a different approach is followed and both convolutional and recurrent neural networks are tested to classify a time series as chaotic or non-chaotic. Works such as [16–18] pursue a LCE-based analysis by combining DNNs with the typical methods for chaos detection: in [17, 18] an empirical data-driven solution was developed to provide chaos detection without any information on the system dynamics; in [16] 2D-bifurcation diagrams were reconstructed with various DNNs to identify chaos over the parameter space of a given system.

Following a successful chaos detection, a typical case study involves the prediction of a chaotic system time evolution, as it may be used to plan/optimize the applied control or the system structure itself. We mention that DNN have found significant applications also in this direction [19–25], as they offer computationally-efficient tools with demonstrated high accuracy.

As mentioned, one of the main limitations of most of the chaos detection approaches available in the literature is due to the natural presence of additive noise in empirical measurements. To limit the impact of noise on chaos detection, works such as [14, 15] focused on discerning purely stochastic time series from trajectories of chaotic systems affected by noise. In [14], several DNN are tested for chaos detection for a total of six chaotic systems, but interestingly the authors trained all DNN as binary classifiers, hence assuming *a priori* knowledge on the system dynamics. We mention that the authors of [14], similarly to [13], also test whether their better-performing DNN is capable of discerning different chaotic dynamics with respect to the one it was specifically trained for, but observed limited accuracy that highlights an overall low generalization. A different approach is followed by the authors of [15], which consider four different chaotic systems corrupted by three types of additive coloured noise (white, pink or brown). Chaos detection is then solved by designing a DNN to classify either the time series

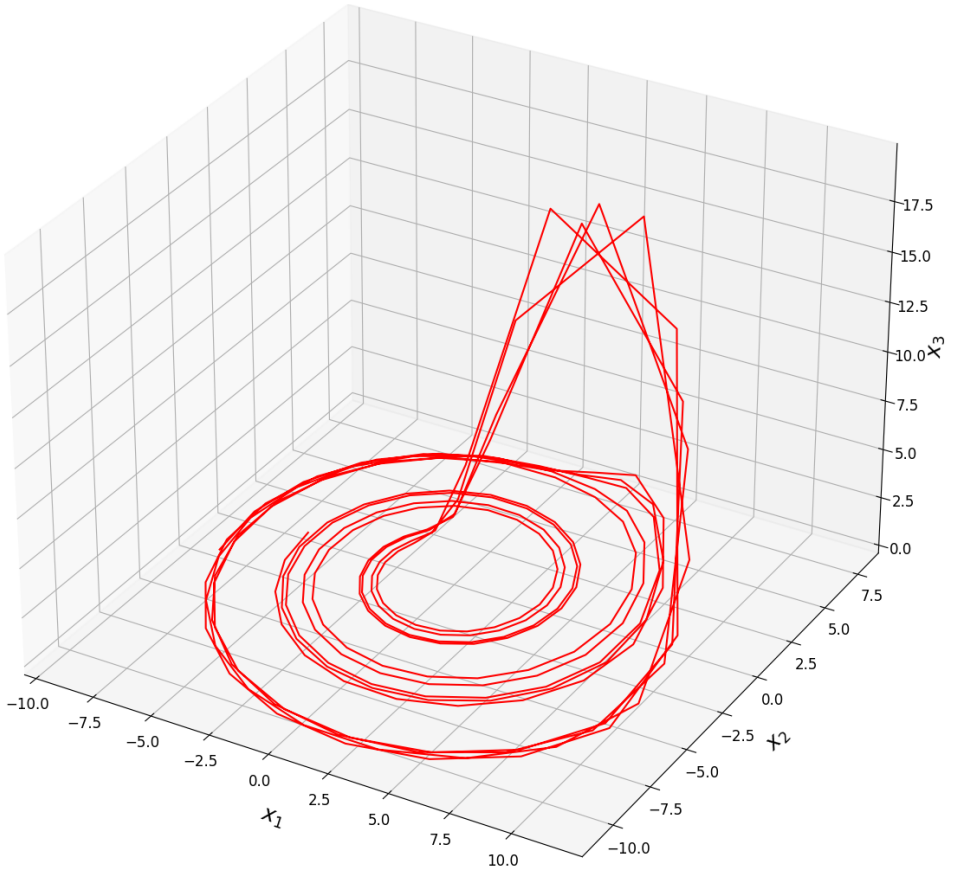

Figure 1. Phase portrait of one of the trajectories generated by an autonomous dissipative flow that is showing a Rössler attractor.

as chaotic or stochastic, identifying also the noise affecting the time series. Similarly to the present paper, in [15] the DNN is fed with recurrence plots and spectrograms, but differently from our approach, the authors discard the time series and employ a single DNN channel and provide no further classification regarding the specific chaotic dynamics that generated the data.

The present paper proposes a specialized DNN capable of first analysing in parallel, and then combining information coming from recurrence plots, spectrograms and time series to solve chaos detection in the presence of significant levels of measurement noise. The DNN has been designed to recognize which specific system dynamics is the most likely to have generated the sampled data among the five chaotic and ten non-chaotic systems presented in [3, 26].

The proposed method is the first one capable of multi-class classification of chaotic systems while also demonstrating robustness against measurement noise, as shown by the simulations of Section 5.

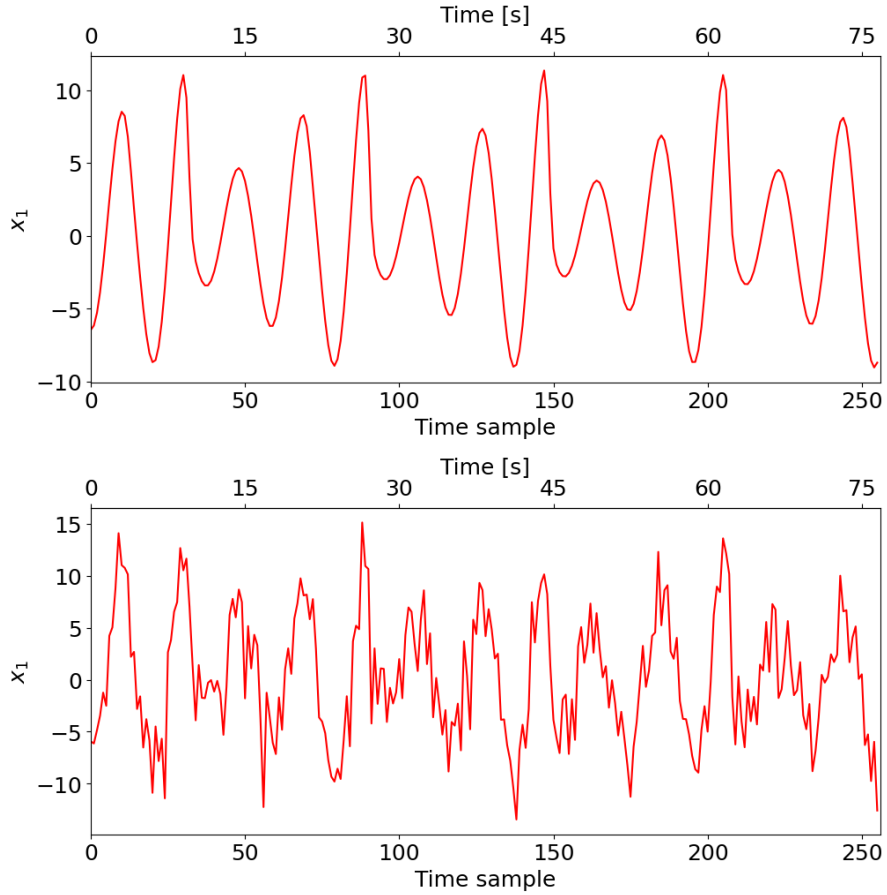

Figure 2. (top) Time series of measurements relative to the x_1 state for the example of Figure 1; (bottom) same time series affected by a white gaussian noise with a Signal-to-Noise Ratio of 5 dB.

3. Preliminaries

This section introduces and provides the definitions of the main tools employed for the analysis of the time series and the design of the proposed DNN.

3.1. Problem Description: Chaos detection in noisy time serie

For the design of DNNs we will adhere to the following assumptions: i) the time series available for our analysis will be limited to a single state component, as in the example in Figures 1 and 2; ii) the system dynamics are sampled from one out of 15 different sets of dynamics.

The former assumption allows us to feed our system with time series sampled from a broad class of systems, independently from their state dimension. We mention that extending the proposed architecture to the multi-dimensional case is almost seamless, as it would be sufficient to either stack equivalent inputs (e.g., all recurrence plots) before feeding them to the convolutional layers or to add some additional channels in the DNN. Also, the number of dynamics is arbitrary and is derived from the dataset

we employed, [26], which is significantly higher than the typical number of dynamics considered in similar studies such as [12, 15].

3.2. Chaotic Dynamics

The five chaotic dynamics considered for our design are derived from [3] and detailed in the following:

(i) Ueda oscillator

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -x_1^3 - bx_2 + Asin(\Omega t), \end{cases} \tag{1}$$

with $\Omega = 1, A = 7.5, b = 0.05$.

(ii) Lorenz attractor

$$\begin{cases} \dot{x}_1 = -\sigma x_1 + \sigma x_2 \\ \dot{x}_2 = \rho x_1 - x_2 - x_1 x_3 \\ \dot{x}_3 = -\beta x_3 + x_1 x_2 \end{cases} \tag{2}$$

with $\sigma = 10, \beta = 8/3, \rho = 28$.

(iii) Rossler attractor

$$\begin{cases} \dot{x}_1 = -x_2 - x_3 \\ \dot{x}_2 = x_1 + ax_2 \\ \dot{x}_3 = b + x_3(x_1 - c) \end{cases} \tag{3}$$

with $a = 0.2, b = 0.2, c = 5.7$.

(iv) Halvorsen attractor

$$\begin{cases} \dot{x}_1 = -ax_1 - 4x_2 - 4x_3 - {x_2}^2 \\ \dot{x}_2 = -4x_1 - ax_2 - 4x_3 - {x_3}^2 \\ \dot{x}_3 = -4x_1 - 4x_2 - ax_3 - {x_1}^2 \end{cases} \tag{4}$$

with $a = 1.27$.

(v) Rucklidge attractor

$$\begin{cases} \dot{x}_1 = -kx_1 + \lambda x_2 - x_2 x_3 \\ \dot{x}_2 = x_1 \\ \dot{x}_3 = -x_3 + x_2^2 \end{cases} \tag{5}$$

with $k = 2, \lambda = 6.7$.

Regarding the non-chaotic dynamics considered, [3] includes a broad range of dynamical systems with periodic, quasi-periodic and non-periodic behaviours.

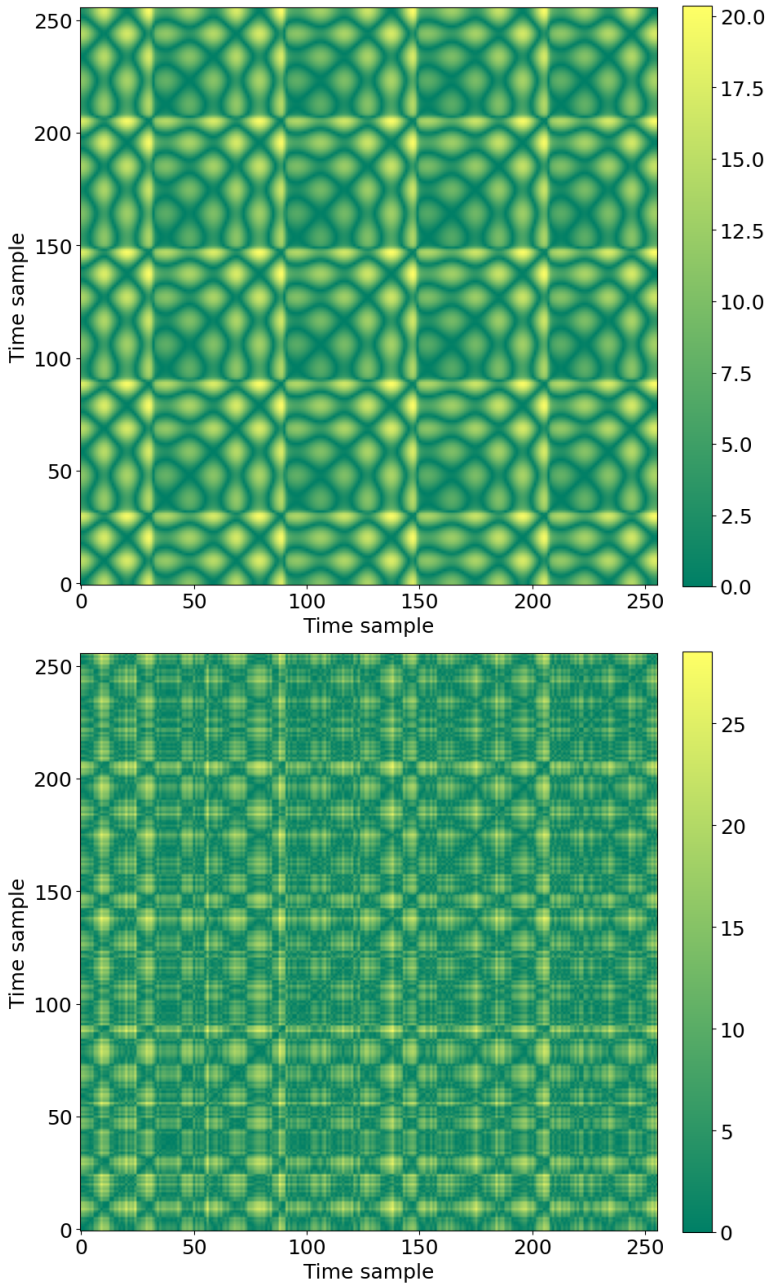

Figure 3. Unthresholded recurrence plot of the time series of Figure 2 (top) and of the same time series affected by pink noise with a SNR of 1 dB (bottom).

3.3. Recurrence Plots

Recurrence plots [27] were originally presented in the 80's to provide a clear visualization of how a system trajectory evolves in its phase space in terms of how often it *recurs* in a neighbourhood of a state it previously visited. Let $x(t)$ be the state of a dynamical system at time t; a trajectory of state measures $x(1), x(2), ..., x(N)$ is used to define the so-called recurrence matrix $\mathbf{R}$, whose elements are defined as [27]:

$$\mathbf{R}_{i,j} = \begin{cases} 1 & \text{if } x(i) \approx x(j) \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

where the operator $\approx$ is used to represents equality up to a threshold $\epsilon > 0$ in the sense that $||x(i) - x(j)|| < \epsilon$.

A recurrence plot is then a way to visualize graphically, encoding each pixel in black or white, the entries of the matrix (6) and can be used to identify periodic, quasi-periodic and recurrence towards attractor behaviours, depending on the particular structures that appear (e.g., diagonal lines, isolated points, patterns).

Several variations of recurrence plots have been proposed to capture different characteristics of the system dynamics by considering different norms, definitions of the distance among two states $x(i), x(j)$ and/or thresholds [27], impacting on the very definition of the concept of recurrence used in their analysis. This study employs the so-called unthresholded recurrence plots [28] (e.g., Figure 3), also know as distance plots, which enriches the information visualized by (6) by removing the threshold ϵ and instead encoding in a range of colours the distance $||x(i) - x(j)||$, i.e.:

$$\mathbf{R}_{i,j} = ||x(i) - x(j)||. \tag{7}$$

The rationale behind our choice is to provide the DNN with as much information as possible, as the threshold introduced in (6) discards any insight on the distance magnitude and relies on the arbitrarily set ϵ. In fact, providing the DNN with the unthresholded recurrence plot allows it to identify autonomously the most effective features to consider to define recurrence.

From Figure 3 we see how noise affects recurrence plots by apparently blurring them with random vertical and horizontal lines. However, we note how the general location of peaks and valleys is preserved, suggesting that information about recurrence is preserved.

3.4. Spectrograms

Spectrograms are among the most broadly used visualization tools to analyse time series as they show how the spectrum of frequencies of the given signal varies with time.

The typical way of representing a spectrogram is as a heatmap $\mathbf{S}$, as the ones depicted in Figure 4, in which for a given time j the elements/pixels $\mathbf{S}_{i,j}$ encode the power spectral density (in dB) for the various frequencies i.

From Figure 4 it is clear that noise re-distributes power over the various frequencies. Having considered three different coloured noises (white, pink and brown), the effect they cause on the original signal spectrum follows their power distribution, which is uniform for white noise, inversely proportional to frequency for pink noise and inversely proportional to the square of the frequency for brown noise.

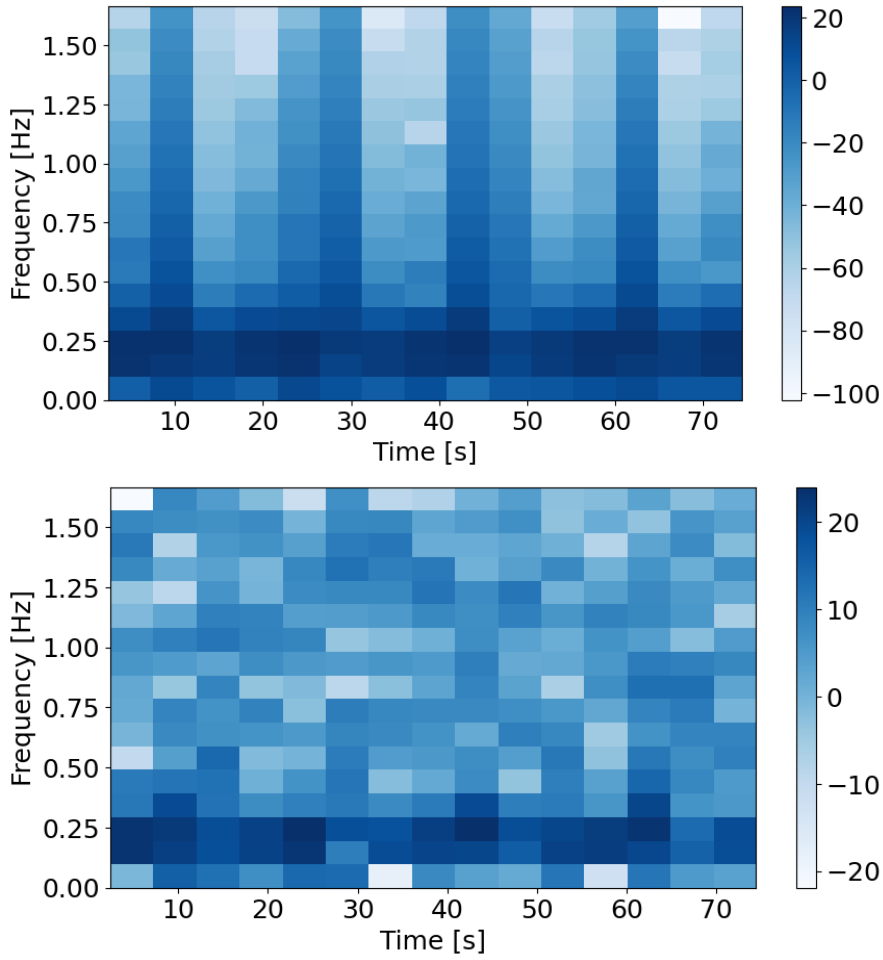

Figure 4. (top) Spectrogram of the trajectory of Figure 2; (bottom) spectrogram of the same time series affected by pink noise with a SNR of 1 dB .

3.5. Multimodal deep neural networks

One of the most significant breakthroughs of deep learning that enabled DNN to reach unprecedented performance in solving complex tasks is the capability of DNNs to conduct their analysis directly on raw, un-processed, data. This peculiarity of DNNs is caused by their ability to learn through the learning process hierarchical representations of their training data, hence conducting an automatic feature extraction process [29].

Despite the well-established results of DNN in terms of automated feature extraction, a significant effort was spent on designing specialized DNN architectures able to capture more easily certain features and patterns in different data structures (e.g., convolutional DNN to recognize geometrically-close patterns in images). In this direction, multimodal DNN [30] are a class of DNN specialized to deal with heterogeneous input data, that is, data of various structures/*modalities* (e.g., the combination of images, signal readings and text). In multimodal DNNs, the first layers of the DNN are replaced by multiple separate *channels* that analyse individually and in parallel the various inputs. This separate processing allows the employment of different

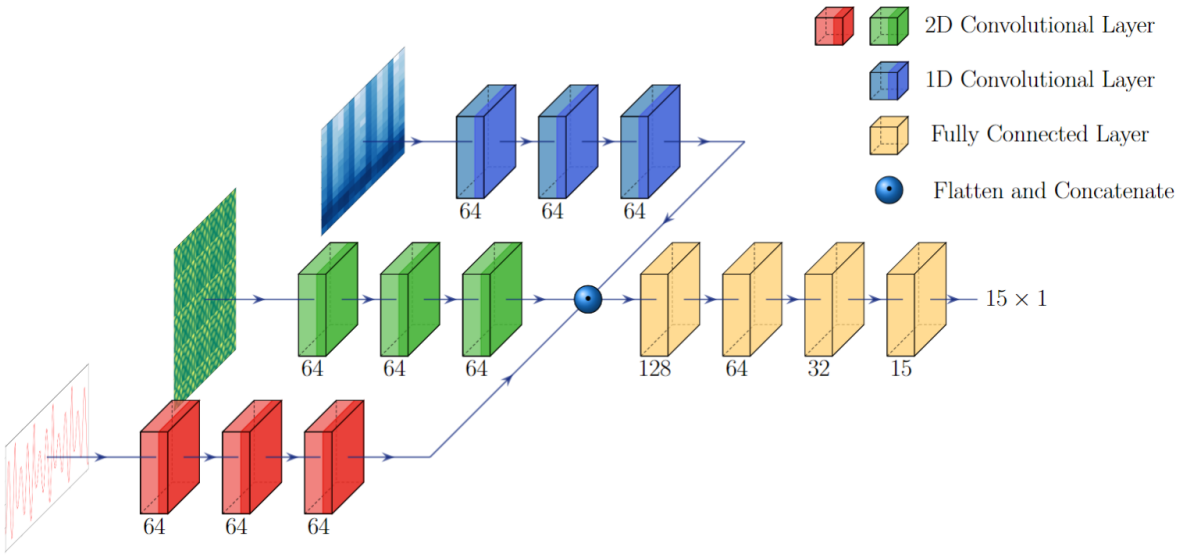

Figure 5. Multimodal deep neural network architecture for chaos detection.

specialized architectures on the various channels, hence carrying out a more effective feature extraction on each modality.

The extracted features are then combined in the deeper layers of the DNN, allowing for a more comprehensive, yet less complex, analysis of the heterogeneous data in its entirety.

4. Proposed multimodal deep neural network

In this section we detail the multimodal DNN architecture and its training process.

4.1. DNN Architecture

The proposed multimodal architecture is depicted in Figure 5 and consists of three separate *convolutional channels* that converge into a common set of *fully connected* (FC) layers, which ends with a 15-neuron layer which implement a softmax function to perform the classification among the 15 different dynamics assumed in section 3.1.

Fully connected layers [31] are the simplest element of DNN architectures and are formed by a set of neurons, each connected to every neuron of the previous layer. Having all possible connections, FC layers have a very high capacity (i.e., the degree of complexity of the function they can approximate). Hence, in complex architectures, FC are typically used as final layers to produce the DNN output based on the results obtained by some more specialized layers.

Convolutional layers [31] are the constituting elements of convolutional neural networks and exploit the local connectivity of their neurons, which share the same weights, to implement a sliding-window/convolutional analysis of their input. Due to the geometric nature of the convolution process, convolutional layers have found

vast applications in vision-related tasks and signal processing, serving the purpose of extracting information (or *features*) from an image/signal-like input while preserving the geometric/spatial nature of the data. Convolutional layers are hence typically used as the first layers of a DNN as *feature extractors* for the deeper layers, which are commonly FC.

In our architecture, we employ three different convolutional channels to analyse separately the three different inputs (i.e., a time series, its recurrence plot and its spectrogram). For the recurrence plot and spectrogram channels, since their input consists of images, we used 2D-convolutional layers, whereas 1D-convolutions were used for the time series channel. The output of the three convolutional channels is then flattened and concatenated before being fed to the final, common, FC portion of the DNN. For the sake of completeness, we mention that pooling layers [31], which are the third element commonly found in standard Convolutional DNNs such as LeNets [32] and VGG networks [33], were not included in our architecture. In fact, the main advantage of using max/average pooling layers is to reduce the dimension of the feature maps internally processed by the convolutional neural network by compressing information and, in turn, lower the number of trainable parameters used. Given the reasonably limited number of parameters involved in the training phase of our DNN, which is kept low also thanks to the multi-step training procedure detailed in section 4.2, we did not observe any significant benefit from their inclusion but, in principle, they may be seamlessly included in deeper networks for chaos detection derived from ours.

We remark that, since both the recurrence plots and spectrograms are derived from the measurement time series, the transformations needed to obtain them can be seen as initial feature extractors for the time series, that are further refined by the CNN channels.

For our tests, we set all neuron activation functions to be Rectified Linear Units (ReLU), save for the final layer that used the Softmax activation function as customary in classification tasks.

4.2. Training process

To better capture the characteristics of each data representation, the three channels were trained separately and then combined in a multi-step training procedure. This approach reduces the resources needed for the training, as only a portion of the DNN is trained at a given time, but its main advantage is that it ensures that each convolutional channel produces informative feature maps to be fed to the final FC layers. In fact, since such feature maps were originally used individually to solve the same classification task, their usage allows for a simpler training and reduces the tendency of multimodal DNNs to rely mostly on a single modality [34].

To train the convolutional layers of the three channels represented in Figure 5, we first trained three separate DNNs, each constituted by such convolutional layers followed by three additional FC layers with 64, 32 and 15 neurons. This individualized

training requires the three DNNs to recognize the chaotic dynamics relying only on the single data modality provided (e.g., only on the recurrence plot), meaning that the three convolutional layer stacks become feature extractors specialized for their respective data.

The FC layers are then discarded, and the trained convolutional layers are placed in the multimodal architecture depicted in Figure 5. The multimodal network is then further trained, this time using all data modalities at the same time, keeping *frozen* the previously obtained weights of the convolutional layers and allowing the update of only the layers following the concatenation, as it is done in a transfer learning procedure.

The presented training process is one of the possible approaches to train a multimodal model designed following the *late fusion* paradigm [35–37], as is the case for our DNN. Compared to the others (e.g., ensembling multiple single-modal DNNs) it provides a solution that correlates/fuses the various modalities through an ad-hoc learning process (that is, the training of the final FC layers), while avoiding the more complex task of training directly the multimodal DNN in its entirety, which may be affected by a so-called *greedy learner* behaviour (that is a tendency to rely on a single modality) and may require significantly more hyperparameters tuning. The proposed multi-step training procedure hence allows for better information fusion regarding the analysis of the three data modalities, leading to an overall better performance when compared to single-channel DNNs, as the one used in [15], or multi-model architectures trained directly on multiple input types (as it will be shown in Section 5).Furthermore, we stress that the proposed procedure requires significantly less memory and computing power when compared to the direct training of the entire multimodal DNN, making the system more scalable and allowing to easily expand the DNN with additional channels if needed.

5. Simulations

5.1. Dataset description

As mentioned, the dataset used for our testing has been derived from the one presented in [3] and made available by the authors in [26]. The starting dataset included time series sampled from a total of 15 different dynamical systems, including 5 chaotic systems and 10 systems generating periodic, quasi-periodic and non-periodic signals. For our tests, we considered 1000 trajectories for each of the 15 systems, generating additional ones when needed using the original code from the authors made available in [26], and reserved 25% of data for validation.

As mentioned, we extracted from the dataset only the trajectories reporting the first component of the various systems, from which we extracted time series of 256 samples. Such trajectories were then corrupted by additive random noise so that the resulting Signal-to-Noise-Ratio (SNR) of each trajectory was respectively equal to 5 dB and 1 dB for our first and second tests. Regarding the noise nature, each trajectory was subject with a uniform probability to either white, pink or brown noise, generated

using the Coloured Noise python library [38]. Spectrograms were constructed using the specgram() function from the matplotlib library using 32 points for the fast Fourier transform blocks with 16 points of overlap.

All training was conducted on a machine equipped with an RTX 3090 GPU. All of the three convolutional networks were trained for 15 epochs, which required about 30 seconds each, with an additional 15 epochs reserved for the fine-tuning of the last layers of the multimodal DNN, in line with the procedure of section 4.2.

To identify our classes, we will follow the naming scheme of [3] that uses the following suffixes: CHA = chaotic; QPS = quasi-periodic; OSC/DOSC/IOSC = undamped/damped/rising oscillators; DS = damped systems. Classes were then numbered according to the naming in [3] as: 0 = OSC_1; 1 = DS_1; 2 = OSC_2; 3 = CHA_4 (an autonomous dissipative flow showing a Halvorsen attractor); 4=QPS_2; 5 = CHA_5 (an autonomous dissipative flow showing a Rucklidge attractor); 6=QPS_1; 7 = CHA_3 (an autonomous dissipative flow showing a Rössler attractor); 8=DOSC_1; 9 = DOSC_2; 10=CHA_2 (the Lorenz system); 11 = QPS_3; 12=IOSC; 13=CHA_1 (a driven dissipative flow showing a Ueda oscillator); 14 = DS_2.

5.2. Results

Figure 6 reports the validation confusion matrix, normalized by rows between 0 and 1, of our first test in which we set the SNR to 5 dB. The resulting total accuracy (number of successful predictions made by the model over all the classes divided by the total number of predictions) is about 94%, with most miss-classifications occurring for class 1 that is mistaken for class 14. These mistakes can be explained by considering that classes 1 and 14 both represent damped linear systems that differ mostly by the magnitude of their damping. Figure 7 confirms the good quality of the classification by reporting the F1-score archived by our DNN: on the right side of the figure, every class is given its own class F1-score computed as the harmonic mean of the class precision and recall; on the left portion of the figure, we report the macro-F1 score (that is the arithmetic average of the class F1-scores), the micro-F1 score (that is the harmonic mean of the overall precision and recall of the model) and the weighted-F1 score (that is the arithmetic average of the class scores weighted by their sample number).

For the sake of comparison and to validate the effectiveness of our multi-step training procedure described in section 4.2, we mention that the best-performing single-modal DNN was the one trained on RPs which had an overall accuracy of about 85%, whereas training directly the entire multimodal architecture of Figure 5 led to an overall accuracy of about 90%.

For our second test, we significantly increase the noise power, reaching an SNR of 1 db. Figure 8 highlights how the DNN accuracy drops for all classes, reaching an average accuracy of about 76% (65% for single channels), with the most affected classes being once again 1 and 14. Class 9 is also commonly mistaken for class 8 (both are oscillating dynamics with decreasing amplitude), whereas class 4 (a quasi-periodic

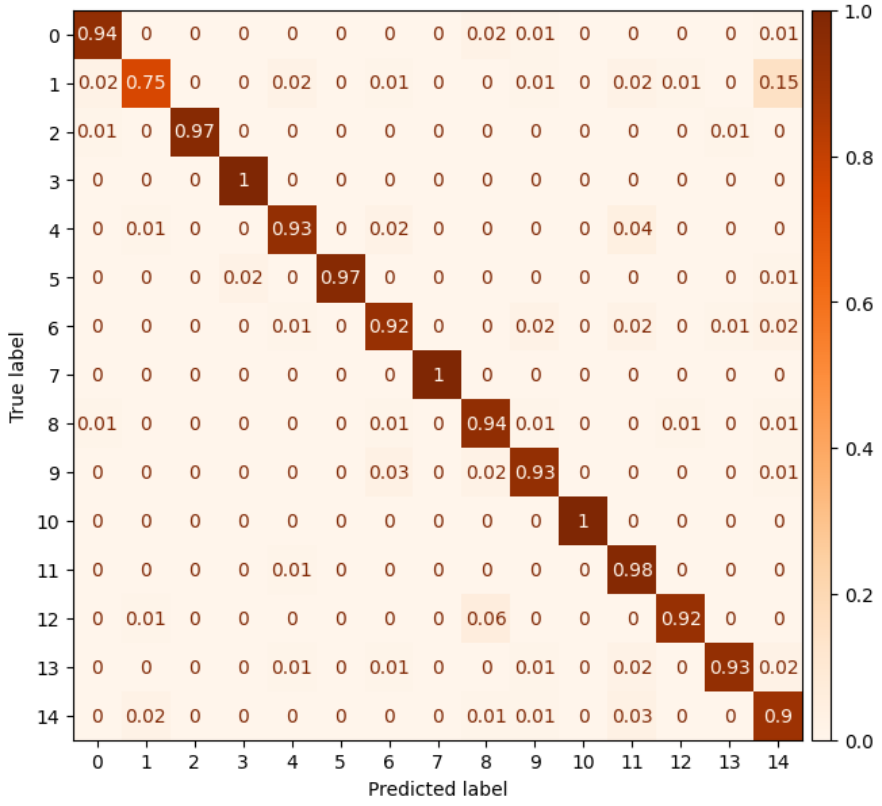

Figure 6. Confusion matrix for the first test (SNR= 5 dB)

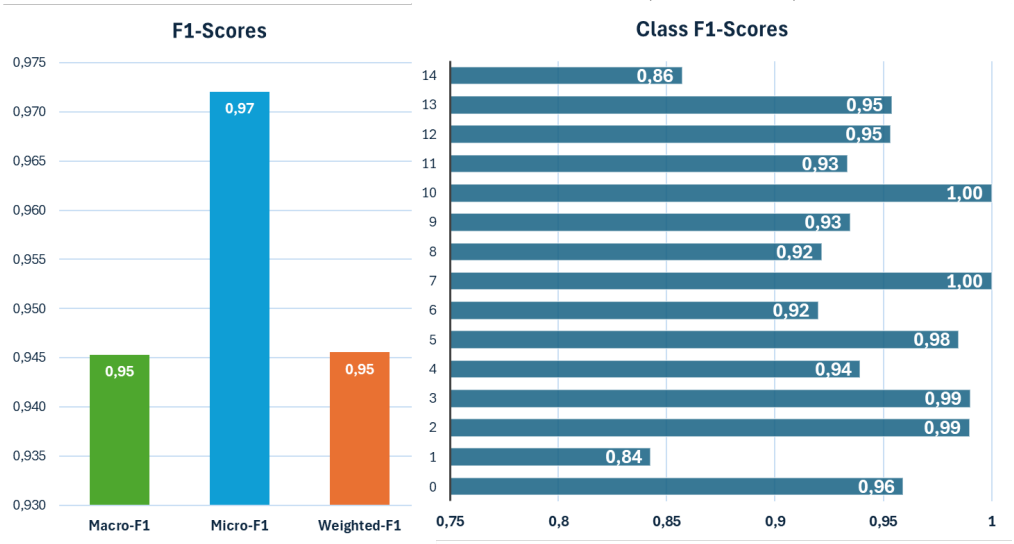

Figure 7. F1-Scores for the first test (SNR = 5 dB)

nonlinear system) is mistaken mostly for the oscillating dynamics of class 8 or for class 6 (that is another quasi-periodic nonlinear system). Overall, we can conclude that as expected the higher level of noise makes the distinction between similar sets of dynamics more complex; however, chaos detection still reaches satisfactory results with the chaos-related classes (3, 5, 7, 10, 13) maintaining an adequate level of performance, as also

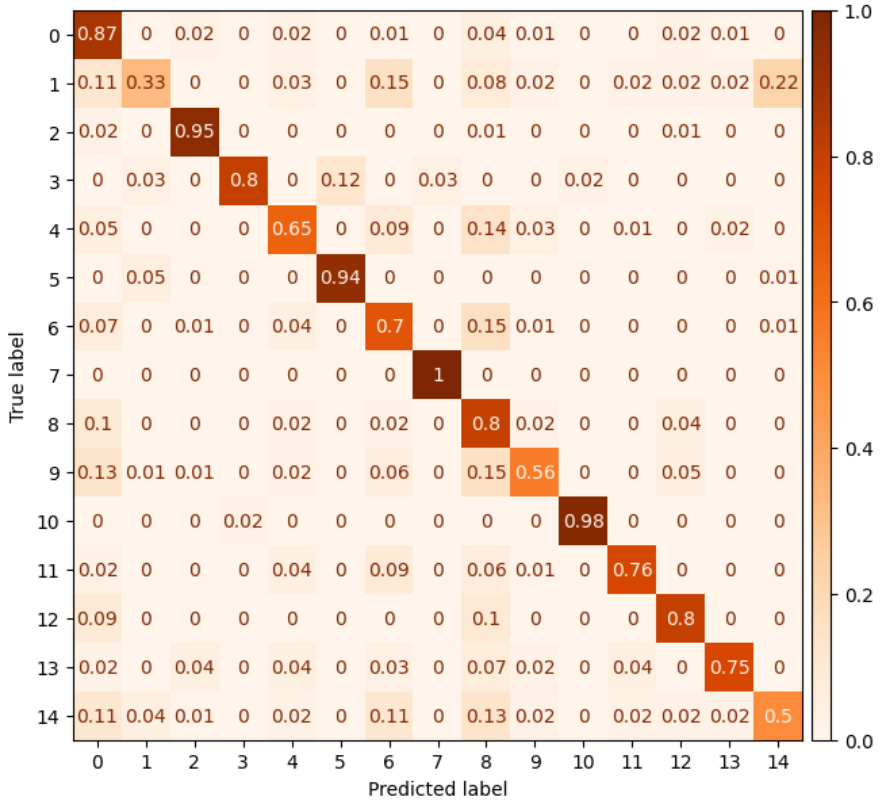

Figure 8. Confusion matrix for the second test (SNR= 1 dB)

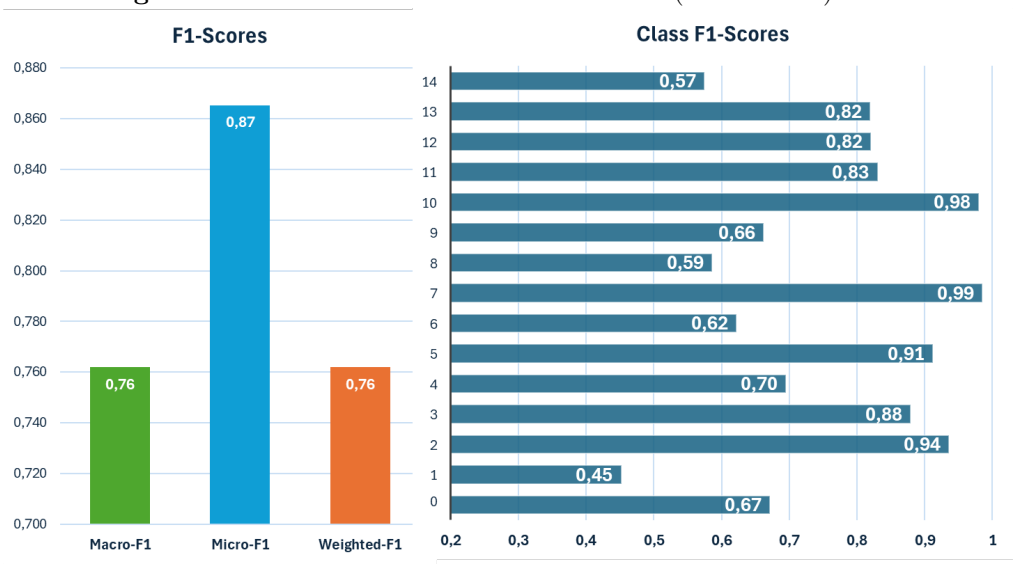

Figure 9. F1-Scores for the second test (SNR = 1 dB)

supported by the F1-scores of Figure 9.

As a final test, we task our DNN with classifying trajectories sampled from new, unseen, dynamics that were not included in the training set. To do so, we generated a new dataset of 1000 trajectories for each of the five dynamics of our chaotic systems. Each of such trajectories was generated by randomly perturbing each of the parameters

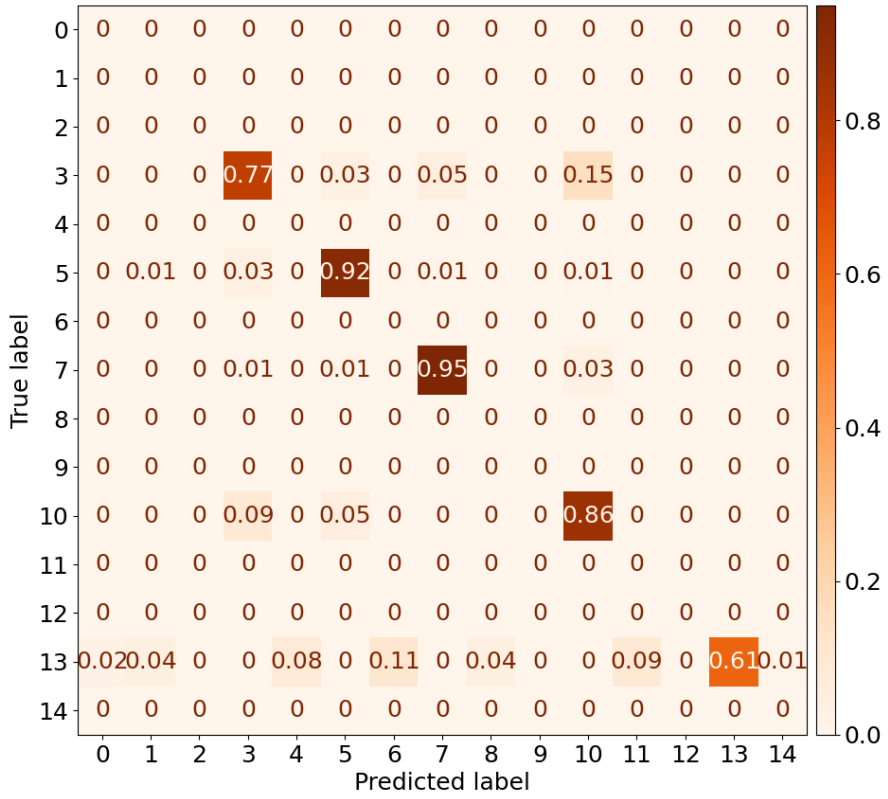

Figure 10. Confusion matrix for the third test (only chaotic systems with different parameters)

of the original dynamics of a factor equal to ±5% of its original values. Figure 10 displays the confusion matrix of such a classification with a snr equal to 1 dB: from the diagonal entries of the matrix it is clear that the perturbed dynamics are harder to recognize, however overall the DNN maintains an accuracy of 82.2%, with a micro-F1 score of 0.90 and macro-F1 score of 0.86, suggesting that the DNN is capable of identifying to a reasonable extend which known dynamics is the most similar to the one observed.

6. Conclusions and Future Works

This work has presented the design of multimodal deep neural network (DNN) for the problem of chas detection in measurements affected by noise. The proposed DNN was designed to solve a multi-class classification problem, with the goal of not only detecting chaotic behaviour but also identifying the specific attractor that characterizes the underlying system dynamics. The ad-hoc architecture of the multimodal DNN we designed involves three parallel input channels, each respectively specialized for the analysis of recurrence plots, spectrograms and time series. The overall DNN was trained following a custom multi-step procedure that consists in first training each channel separately and then training the common portion of the DNN to correctly fuse the information extracted from the various input modalities. Our system was able to

discern among 15 different dynamics from the measurement of a single state component, demonstrating high accuracy in discerning chaotic dynamics from non-chaotic ones even in the presence of significant levels of noise.

Future works are related to increasing the number of dynamics considered by extending the dataset with some other relevant chaotic systems and in extending the capabilities of the proposed DNN. Regarding the multimodal input, we aim at adding more complex and informative inputs/channels into the DNN, whereas regarding the output we are designing a DNN that employs multiple output channels to provide different analysis: a first relevant output to include would be a neural operator capable of estimating the Lyapunov exponent [8], while a more challenging and innovative study would integrate an output channel consisting of a Kolmogorov-Arnold Network (KAN) [39] to provide a symbolic guess of the observed dynamics.

Acknowledgements

This work has been partially supported by the EU in the framework of the FSE REACT-EU, PON Ricerca e Innovazione 2014-2020, programme.

Bibliography

[1] Georg A. Gottwald and Ian Melbourne. On the implementation of the 0–1 test for chaos. *SIAM Journal on Applied Dynamical Systems*, 8(1):129–145, January 2009.

[2] Daniel Toker, Friedrich T. Sommer, and Mark D'Esposito. A simple method for detecting chaos in nature. *Communications Biology*, 3(1), 2020.

[3] Agnieszka Szczesna, Dariusz Augustyn, Katarzyna Harezlak, Henryk Josinski, Adam Switonski, and Pawel Kasprowski. Datasets for learning of unknown characteristics of dynamical systems. *Scientific Data*, 10(1):79, 2023.

[4] Christos H Skiadas and Charilaos Skiadas. *Handbook of applications of chaos theory*. crc Press, 2017.

[5] Erol Basar. *Chaos in Brain Function*. Springer Berlin Heidelberg, 1990.

[6] F Tito Arecchi and Robert G Harrison. *Instabilities and chaos in quantum optics*, volume 34. Springer Science & Business Media, 2012.

[7] Arunachalam Kumar. Chaos theory: impact on and applications in medicine. *Journal of Health and Allied Sciences NU*, 02(04):93–99, 2012.

[8] Henry D. I. Abarbanel, Reggie Brown, and M. B. Kennel. Lyapunov exponents in chaotic systems: their importance and their evaluation using observed data. *International Journal of Modern Physics B*, 05(09):1347–1375, 1991.

[9] Alan Wolf, Jack B. Swift, Harry L. Swinney, and John A. Vastano. Determining lyapunov exponents from a time series. *Physica D: Nonlinear Phenomena*, 16(3):285–317, 1985.

[10] Georg A. Gottwald and Ian Melbourne. *The 0-1 Test for Chaos: A Review*, page 221–247. Springer Berlin Heidelberg, 2016.

[11] Roberto Barrio, Alvaro Lozano, Ana Mayora-Cebollero, Carmen Mayora-Cebollero, Antonio Miguel, Alfonso Ortega, Sergio Serrano, and Rubén Vigara. Deep learning for chaos detection. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 33(7), 2023.

[12] Nicolas Boullé, Vassilios Dallas, Yuji Nakatsukasa, and D. Samaddar. Classification of chaotic time series with deep learning. *Physica D: Nonlinear Phenomena*, 403:132261, 2020.

[13] Woo Seok Lee and Sergej Flach. Deep learning of chaos classification. *Machine Learning: Science & Technology*, 1(4):045019, 2020.

[14] Massimiliano Zanin. Can deep learning distinguish chaos from noise? numerical experiments and general considerations. *Communications in Nonlinear Science and Numerical Simulation*, 114:106708, 2022.

[15] Sumona Mukhopadhyay and Santo Banerjee. Learning dynamical systems in noise using convolutional neural networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(10), 2020.

[16] Salama Hassona, Wieslaw Marszalek, and Jan Sadecki. Time series classification and creation of 2d bifurcation diagrams in nonlinear dynamical systems using supervised machine learning methods. *Applied Soft Computing*, 113:107874, December 2021.

[17] Hagai Rappeport, Irit Levin Reisman, Naftali Tishby, and Nathalie Q. Balaban. Detecting chaos in lineage-trees: A deep learning approach. *Physical Review Research*, 4(1), March 2022.

[18] Dagobert Wenkack Liedji, Jimmi Hervé Talla Mbé, and Godpromesse Kenné. Chaos recognition using a single nonlinear node delay-based reservoir computer. *The European Physical Journal B*, 95(1), January 2022.

[19] Huawei Fan, Junjie Jiang, Chun Zhang, Xingang Wang, and Ying-Cheng Lai. Long-term prediction of chaotic systems with machine learning. *Physical Review Research*, 2(1), March 2020.

[20] Salim Lahmiri and Stelios Bekiros. Cryptocurrency forecasting with deep learning chaotic neural networks. *Chaos, Solitons & Fractals*, 118:35–40, January 2019.

[21] Rui Wang, Eugenia Kalnay, and Balakumar Balachandran. Neural machine-based forecasting of chaotic dynamics. *Nonlinear Dynamics*, 98(4):2903–2917, July 2019.

[22] Rohan Thavarajah, Xiang Zhai, Zheren Ma, and David Castineira. Fast modeling and understanding fluid dynamics systems with encoder–decoder networks. *Machine Learning: Science and Technology*, 2(2):025022, March 2021.

[23] José de Jesús Serrano-Pérez, Guillermo Fernández-Anaya, Salvador Carrillo-Moreno, and Wen Yu. New results for prediction of chaotic systems using deep recurrent neural networks. *Neural Processing Letters*, 53(2):1579–1596, March 2021.

[24] Matteo Sangiorgio, Fabio Dercole, and Giorgio Guariso. Forecasting of noisy chaotic systems with deep neural networks. *Chaos, Solitons &; Fractals*, 153:111570, December 2021.

[25] Shuang Zhou, Zhipeng Zhao, and Xingyuan Wang. Novel chaotic colour image cryptosystem with deep learning. *Chaos, Solitons &; Fractals*, 161:112380, August 2022.

[26] D. Augustyn. Dataset for learning of unknown characteristics of dynamical systems - code, 2023. https://gitlab.com/draugustyn/signal-data.

[27] N Marwan, M Carmenromano, M Thiel, and J Kurths. Recurrence plots for the analysis of complex systems. *Physics Reports*, 438(5–6):237–329, 2007.

[28] Joseph S. Iwanski and Elizabeth Bradley. Recurrence plots of experimental data: To embed or not to embed? *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 8(4):861–871, 1998.

[29] Thomas Wiatowski and Helmut Bolcskei. A mathematical theory of deep convolutional neural networks for feature extraction. *IEEE Transactions on Information Theory*, 64(3):1845–1866, 2018.

[30] Dhanesh Ramachandram and Graham W. Taylor. Deep multimodal learning: A survey on recent advances and trends. *IEEE Signal Processing Magazine*, 34(6):96–108, 2017.

[31] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[32] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014. arXiv:1409.1556.

[34] Nan Wu, Stanislaw Jastrzebski, Kyunghyun Cho, and Krzysztof J Geras. Characterizing and overcoming the greedy nature of learning in multi-modal deep neural networks. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato,

editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 24043–24055. PMLR, 17–23 Jul 2022.

[35] Kuan Liu, Yanen Li, Ning Xu, and Prem Natarajan. Learn to combine modalities in multimodal deep learning, 2018. arXiv:1805.11730.

[36] Konrad Gadzicki, Razieh Khamsehashari, and Christoph Zetzsche. Early vs late fusion in multimodal convolutional neural networks. In *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*. IEEE, July 2020.

[37] Andreas Eitel, Jost Tobias Springenberg, Luciano Spinello, Martin Riedmiller, and Wolfram Burgard. Multimodal deep learning for robust rgb-d object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, September 2015.

[38] John Burkardt. Colored noise $1/f^{\alpha}$ power law noise generation. https://people.math.sc.edu/Burkardt/cpp_src/colored_noise/colored_noise.html.

[39] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y. Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks, 2024. arXiv:2404.19756.