

Analysis of chaotic dynamical systems with autoencoders

Cite as: Chaos **31**, 103109 (2021); <https://doi.org/10.1063/5.0055673>

Submitted: 01 May 2021 • Accepted: 10 September 2021 • Published Online: 05 October 2021

 N. Almazova,  G. D. Barmparis and  G. P. Tsironis

COLLECTIONS

Paper published as part of the special topic on [In Memory of Vadim S. Anishchenko: Statistical Physics and Nonlinear Dynamics of Complex Systems](#)



View Online



Export Citation



CrossMark

ARTICLES YOU MAY BE INTERESTED IN

[Combining machine learning and data assimilation to forecast dynamical systems from noisy partial observations](#)

Chaos: An Interdisciplinary Journal of Nonlinear Science **31**, 101103 (2021); <https://doi.org/10.1063/5.0066080>

[Synchronization of nonlinearly coupled networks based on circle criterion](#)

Chaos: An Interdisciplinary Journal of Nonlinear Science **31**, 103110 (2021); <https://doi.org/10.1063/5.0055814>

[Dynamical ergodicity DDA reveals causal structure in time series](#)

Chaos: An Interdisciplinary Journal of Nonlinear Science **31**, 103108 (2021); <https://doi.org/10.1063/5.0063724>

APL Machine Learning

Open, quality research for the networking communities

MEET OUR NEW EDITOR-IN-CHIEF

LEARN MORE



Analysis of chaotic dynamical systems with autoencoders

Cite as: Chaos 31, 103109 (2021); doi: 10.1063/5.0055673

Submitted: 1 May 2021 · Accepted: 10 September 2021 ·

Published Online: 5 October 2021




View Online



Export Citation



CrossMark

N. Almazova,¹  G. D. Barmparis,^{1,2,a)}  and G. P. Tsironis^{1,2} 

AFFILIATIONS

¹National University of Science and Technology MISiS, Leninsky prosp. 4, Moscow 119049, Russia

²Institute of Theoretical and Computational Physics and Department of Physics, University of Crete, P.O. Box 2208, 71003 Heraklion, Greece

Note: This paper is part of the Focus Issue, In Memory of Vadim S. Anishchenko: Statistical Physics and Nonlinear Dynamics of Complex Systems.

a) Author to whom correspondence should be addressed: barmparis@physics.uoc.gr

ABSTRACT

We focus on chaotic dynamical systems and analyze their time series with the use of autoencoders, i.e., configurations of neural networks that map identical output to input. This analysis results in the determination of the latent space dimension of each system and thus determines the minimal number of nodes necessary to capture the essential information contained in the chaotic time series. The constructed chaotic autoencoders generate similar maximal Lyapunov exponents as the original chaotic systems and thus encompass their essential dynamical information.

Published under an exclusive license by AIP Publishing. <https://doi.org/10.1063/5.0055673>

The quantitative study of complex systems may be assisted by the use of methods derived from neural networks (NNs). The analysis of chaotic time series presents an applied challenge toward determining the features involved and particularly the effective complexity they include. This work shows that the use of chaotic autoencoders made of neural networks leads to essential information on the time series and the chaotic systems they arise from. The autoencoders we construct identically map the input chaotic time series to the output. The process passes through the latent space that acts as an information bottleneck for this reconstruction. We estimate the dimension of the latent space numerically and find it smaller than possible embedding dimensions of the reconstruction. The dimension of the latent space hints toward the complexity of the dynamical system underlying the time series. Furthermore, we show that the constructed chaotic autoencoders produce maximal Lyapunov exponents that are very close to those obtained directly from the equations of motion of the chaotic systems. These features show that the constructed chaotic autoencoders are faithful representations of the time series generating dynamical systems, and through their latent space dimension we obtain an estimate of their complexity.

I. INTRODUCTION

Nonlinear dynamical systems, either continuous or discrete, may produce chaos, i.e., irregular evolution similar to probabilistic, stochastic motion. Continuous autonomous chaotic systems are described by at least three coupled ordinary nonlinear differential equations that, when solved numerically, result in trajectories with irregular features. In general non-Hamiltonian cases, the trajectory may fall on a strange attractor that is a chaotic yet distinct structure in the system phase space. Understanding the complex dynamics in phase space is significant especially in the mathematical analysis of the inverse problem. In the latter, we are given a time series of discrete data and we want to find whether this set derives from a chaotic dynamical system or is purely stochastic.^{1–4} This is a significant problem that stems from the need to understand experimental data as well as other data that are readily available in our times. In this problem, we are given a time series of data and we are interested in reconstructing the phase space of the associated dynamical system.^{5–7} The methods that have been developed over the last 30 or more years work quite well and the area is mature. Nevertheless, since these methods are necessarily cumbersome and involve additionally empirical inference, it would be interesting to experiment with the use of methods derived from Artificial Intelli-

gence (AI). More specifically, Machine Learning (ML) provides an ensemble of well tested methods that may assist or even surpass traditional methods of analysis of chaotic systems.^{8,9} The basic aim of the present work is to use unsupervised learning methods and attempt to reconstruct the dynamical features of the data without relying on traditional methods.

We propose an approach that employs autoencoders, a ML tool that is used widely, as a dimensionality reduction technique. Autoencoders are implemented with neural networks and are in effect unit projection operators where the output is as close as possible to the input. In the process of this unit projection, data pass through the latent space where they suffer dimensionality reduction. The effective dimension of the latent space gives the degree of possible data compression and thus eliminates original dimension redundancy. Since the neural network uses fewer nodes in latent space than the number of input nodes, we accomplish compression of the original data. This packing allows detecting the basic information contained in the data. This compression technique involves data regularization in latent space that is responsible for the selection of the optimal number of operating nodes. This strategy allows us to find the essential number of nodes for the trained network that recaptures the complexity of the original time series.

Once the minimal configuration in latent space is found, we need to compare the chaotic autoencoder with the original system. This is accomplished through the comparison of the corresponding maximal Lyapunov exponents, i.e., that of the original trajectories of the system with the one calculated from the reconstructed trajectories generated through the autoencoder. The Lyapunov exponents are a general attribute of nonlinear dynamical systems as they allow us to distinguish the local stability and invariant sets in a particular direction of attractors. The largest Lyapunov exponent (LLE) detects the mean divergence of nearest neighbors in two initial nearby trajectories. The positive value of the exponent defines the degree of chaos in a particular system. There exist different methods for finding the largest Lyapunov exponent.^{10–14} In this work, we use the largest Lyapunov exponent as a measure of the level of chaos resulting from neural network chaotic autoencoders.

Machine learning techniques are used in areas far beyond computer science. Neural networks (NNs) are applied for a wide range of tasks that permit solving problems in engineering as well as science while deep learning expands the ability of generalization, interpolation, etc.^{9,15} Artificial intelligence helps to solve tasks through appropriate data handling but nevertheless, they cannot give deep insight into the solution processes they use. They operate by allowing the computers to learn the features of data and subsequently understand the hierarchies involved. Several machine learning techniques, such as the reservoir computing,^{16,17} the long short-term memory¹⁸ with observers (O-LSTM),¹⁹ have been used to investigate the time evolution of complex systems when the time series is known.^{20–22} Autoencoders are also used in the physical systems due to their main advantage of dimensionality reduction.^{8,23} An autoencoder is a neural network that maps the input data to the output.²⁴ It presents a method of unsupervised learning with the compression of data occurring in the latent space that forms a kind of information bottleneck, due to the smaller number of nodes it includes. The target data for an autoencoder are the input data. Traditionally, autoencoders are used as dimensionality reduction techniques and

salient features of learning. The construction of an autoencoder includes two main parts. The first part is an encoder function, $h(x)$, that compresses the input information, x , while the second part is the decoder, $g(h)$, that takes as input the compressed information and by decompressing it reconstructs the initial input of the encoder. The output function can be written as $\hat{x} = g(h(x))$, where the input sequence should be learned “perfectly.” In other words, the autoencoder attempts to copy the data to the output while first, they pass through a compression bottleneck. The learning results are obtained through a minimization of the appropriate loss function.

In this article, we focus on the construction of chaotic autoencoders. In Sec. II, we detail features of dynamical systems such as phase space reconstruction and largest Lyapunov exponents and report on the methods for their calculation. Subsequently, we define the chaotic autoencoders for three prototypical dynamical systems and show how the information compression operates. We then show how the Lyapunov exponents are evaluated in the chaotic autoencoders, and finally, we conclude.

II. DESCRIPTION OF THE CHAOTIC SYSTEMS

A deterministic chaotic signal behaves as something intermediate between a regular and a stochastic motion. The nonlinear aspect of this behavior of the signal makes difficult the prediction and classification of the physical system, from which the motion comes.^{6,25–27} A chaotic motion produces specific structures in phase space. Systems of differential equations with three or more degrees of freedom may show chaos in their evolution in real or phase space. The number of degrees of freedom of a system describes the number of the necessary independent variables to specify the dynamical state of the system. Chaotic systems are described through their time-evolution equations, the values of their parameters, and the selected initial conditions. We may express the ordinary differential equations guiding the evolution of a state using a variable $x(t) = [x_1(t), x_2(t), \dots, x_i(t)]$, where $x_i(t)$ is the variable describing the i th degree of freedom at time, t , as

$$\frac{dx(t)}{dt} = f(x(t)), \quad (1)$$

where f describes the nonlinear function that characterizes the equations of motion of the system. Given the system of differential equations guiding a dynamical system, we can solve them numerically and follow its evolution in time. Many dynamical systems have been investigated using this methodology. Here, we focus on prototypical systems well known in the literature of chaotic dynamics, viz., the Rössler system and the Lorenz system. Since we want to have a more general idea of the effectiveness of our method, we use both the Lorenz63 system that contains three degrees of freedom and the Lorenz96 one with multiple dependent variables and external forcing. We will describe these systems below.

A. Delay-coordinate embedding

A physical system can only be measured discretely in time, with a given time step, and thus it can be represented as a time series of data. In the case of a chaotic system, we can use several methods

to determine the features of the chaotic data. One of these methods is the time-delay embedding or reconstruction scheme.^{5,26,28} This method allows for the recovery of the entire multidimensional observation in state-space based on the time series of just one of its variables. Let us assume that a time series, X , represents a trajectory of a physical system recorded with a fixed time step as an N -points time series. The reconstructed trajectory, X , can be represented as an $(M \times m)$ matrix,

$$X = [X_1, X_2, \dots, X_M]^T, \quad (2)$$

where $M = N - (m - 1)\tau$, m is an embedding dimension, τ is a time lag (delay), and X_i is a vector representing the state of the system at discrete time i , given by

$$X_i = [x_i, x_{i+\tau}, \dots, x_{i+(m-1)\tau}], \quad (3)$$

with x_i being the i th data point of the time series. In order to find the embedding dimension for a dynamical system, one can implement the Takens theorem.²⁹ Takens introduced the idea that a complete part of dynamics with basic dimension d_s might be observed in an embedding space if the latter has $2d_s + 1$ or higher dimensions. There are a number of practical implementations of Takens theorem although these methods are not necessarily universal.^{3,30,31} In the present work, we use a time lag equal to one and an appropriate embedding dimension that satisfies the Takens theorem.

III. METHODOLOGY

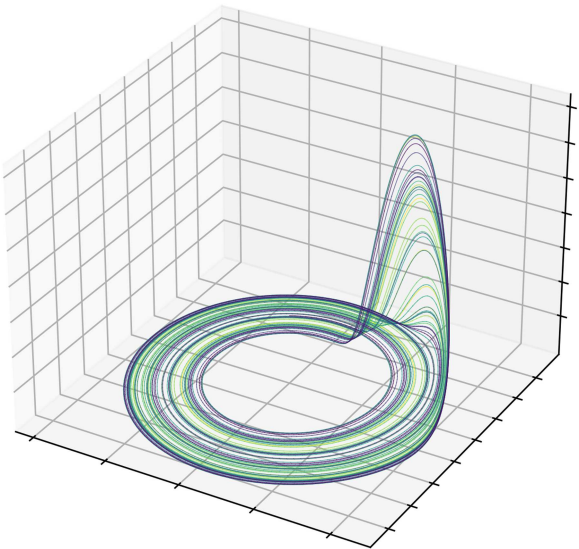
A. Data preparation

In this work, we analyze three chaotic systems, viz., that of Rössler, Lorenz63, and Lorenz96. For the sake of brevity, we present the details of each system in Table I. We integrate the differential equations of each system by means of the fourth order Runge–Kutta method, using a fixed time step equal to 0.005, for 300 000 time steps. The first 50 000 time steps of each solution were removed as transient points. We use the x -coordinate of the Rössler and the Lorenz63 systems, and the 19th coordinate (out of forty) of the Lorenz96 as the training sequences. The 250 000 time steps of each training sequence are normalized between zero and one and split into training (80%) and testing (20%) sets. The training and testing sets are split into smaller pieces using a sliding window of a given size, W .

B. Machine learning: Setting up the chaotic autoencoder

A stacked autoencoder composed of two symmetric networks, the encoder, $h(x)$, and the decoder, $g(h)$, is used to reconstruct each one of the chaotic systems in this work. The encoder consists of an input layer with 30 nodes, i.e., equal to the window size, W , used to split the input sequences. Then, two fully connected hidden layers of 22 and 15 nodes with a Sigmoid and ReLU activation function, respectively, guide the model to the latent space. The latent space is represented by a fully connected layer of ten nodes and a ReLU activation function. L1 regularization (Lasso regression) is applied

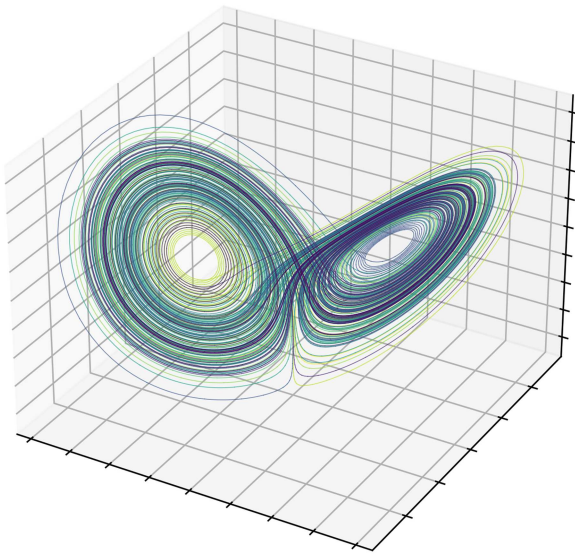
TABLE I. A graphical representation, the equations, and the parameters of each one of the chaotic systems in consideration.

System	Equations	Parameters
Rössler	 $\begin{aligned} \frac{dx}{dt} &= y - z \\ \frac{dy}{dt} &= x + ay \\ \frac{dz}{dt} &= b + z(x - c) \end{aligned}$	$a = 0.1$ $b = 0.1$ $c = 14$

Continued.

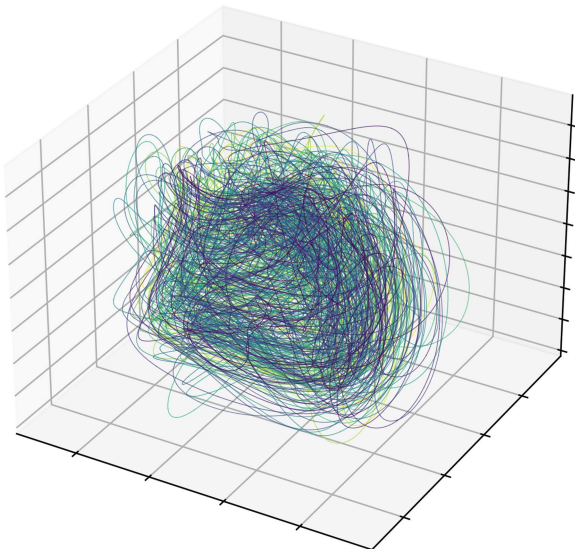
TABLE I. Continued.

Lorenz63



$$\begin{aligned}\frac{dx}{dt} &= \sigma(y - x) & \sigma &= 10 \\ \frac{dy}{dt} &= x(\rho - z) - y & \rho &= 28 \\ \frac{dz}{dt} &= xy - \beta z & \beta &= 8/3\end{aligned}$$

Lorenz96



$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} - x_i + F \quad \begin{aligned} N &= 40 \\ F &= 8.15 \end{aligned}$$

to the output of the nodes of the latent space to force nodes with negligible information to zero. The L1 regularization applied to the output of a layer adds to the loss function a value proportional to the regularization parameter, α , to shrink nodes with less information. The regularization parameter, α , is used as a hyper-parameter of the model. The larger the regularization parameter, the more we constrain the autoencoder and thus we increase the error in the reconstructed sequence. The smaller the regularization parameter, the larger the active part of the latent space and thus we decrease the

error in the reconstructed sequence. After finishing with the training process, we statistically analyze the contribution of the number of active nodes in the latent space to the accuracy of the reconstructed sequence. Nodes that output values smaller than the regularization parameter contain non-important information and their contribution to the reconstructed sequence is negligible. Nodes with values larger than the value of α are counted as non-zero nodes with significant contribution to the output. The decoder network is constructed as the mirror image of the encoder one. The autoencoder model is

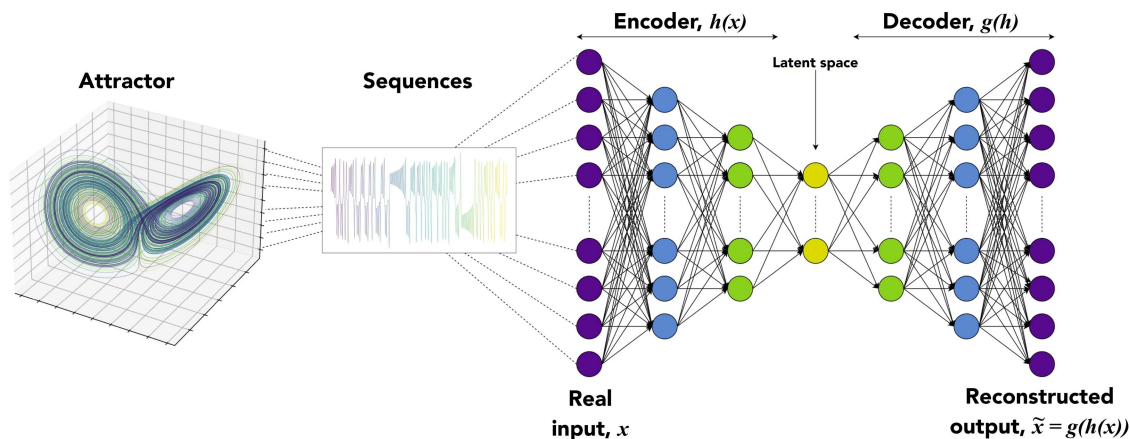


FIG. 1. A schematic representation of the proposed model. The trajectories of a given attractor are transformed into input sequences, x , that pass through an encoder model, $h(x)$, become compressed in the latent space and pass through the decoder, $g(h)$, in order to reconstruct the original information, $\tilde{x} = g(h(x))$.

trained for 7500 epochs with a batch size equal to 32, using the Adam stochastic optimization method³² with a learning rate of 0.001, while all its weights are initialized using the Xavier uniform initializer,³³ as implemented in Tensorflow/Keras.^{34,35} In Fig. 1, we present a schematic representation of the proposed model.

As mentioned in Sec. I, an autoencoder is a neural network that acts as an identity operator and maps the input data ideally to identical output data. In performing this operation through the latent space, it must learn via minimizing a loss function. The simplest example of a loss function is that of the mean squared error (MSE), i.e.,

$$MSE(x, \hat{x}) = \frac{1}{W} \sum_{k=1}^W (x_k - \hat{x}_k)^2, \quad (4)$$

where W is the length of the input sequence, x , represents the real-input data (a set of x_k real samples), $\hat{x} = g(h(x))$ represents the reconstructed-output data (a set of \hat{x}_k reconstructed samples) of a general ML system, while the loss function “punishes” the difference between x and \hat{x} , i.e., the real and reconstructed data. Here, to train the autoencoder, we additionally apply the sparsity penalty (L1 regularization), $\Omega(h) = \alpha|h(x)|$, on the output of the encoder, $h(x)$ (after the activation function), where α , is the regularization parameter. Then, the loss function, L , is represented as

$$L(x, \hat{x}) = MSE + \Omega(h(x)). \quad (5)$$

This feature permits the autoencoder to limit the number of nodes at the latent space to a minimum required number that can be used to reconstruct the original signal correctly. This type of regularization forces the participation of nodes in latent space to be zero when their contribution to the output of the latent space is less than the regularization parameter. Given the fact that we want to investigate the performance of the model as a function of the regularization parameter, α , in order to demonstrate the effect of the regularization and find the minimum number of nodes that can correctly reconstruct the input sequences, we fix all the other parameters of our model and we train it only once. Thus, we can compare the results for the

different values of the regularization parameter. In Fig. 2, we present the evolution of the loss function during the training process for the training and the test set of the Lorenz63 system.

C. Determining α -dependent latent space dimension

The regularization parameter, α , controls the size of the latent space and plays a crucial role in determining the number of the important nodes in the latent space. In our approach, we define as important the nodes in the latent space that output a value larger than the regularization parameter, α . Thus, we calculate the average of non-zero nodes in the test set as a function of α . The results for the three systems in consideration are shown in Fig. 3.

D. Largest Lyapunov exponent as a metric

The Lyapunov exponents are an essential concept of nonlinear dynamics and present specific features of chaotic systems. Two trajectories of the attractor initiated at points displaced infinitesimally

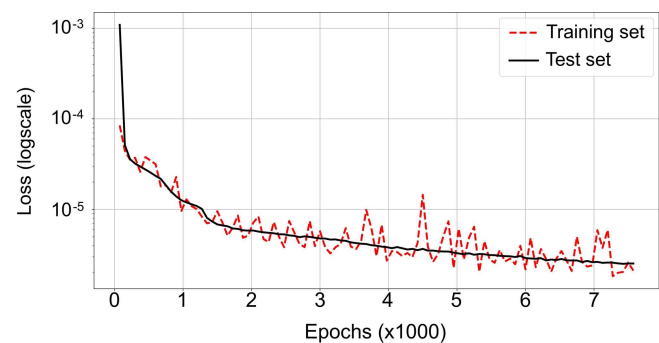


FIG. 2. The evolution of the training (red dashed line) and the test set (solid black line) loss in logarithmic scale, as a function of the number of epochs of training for the Lorenz63 system.

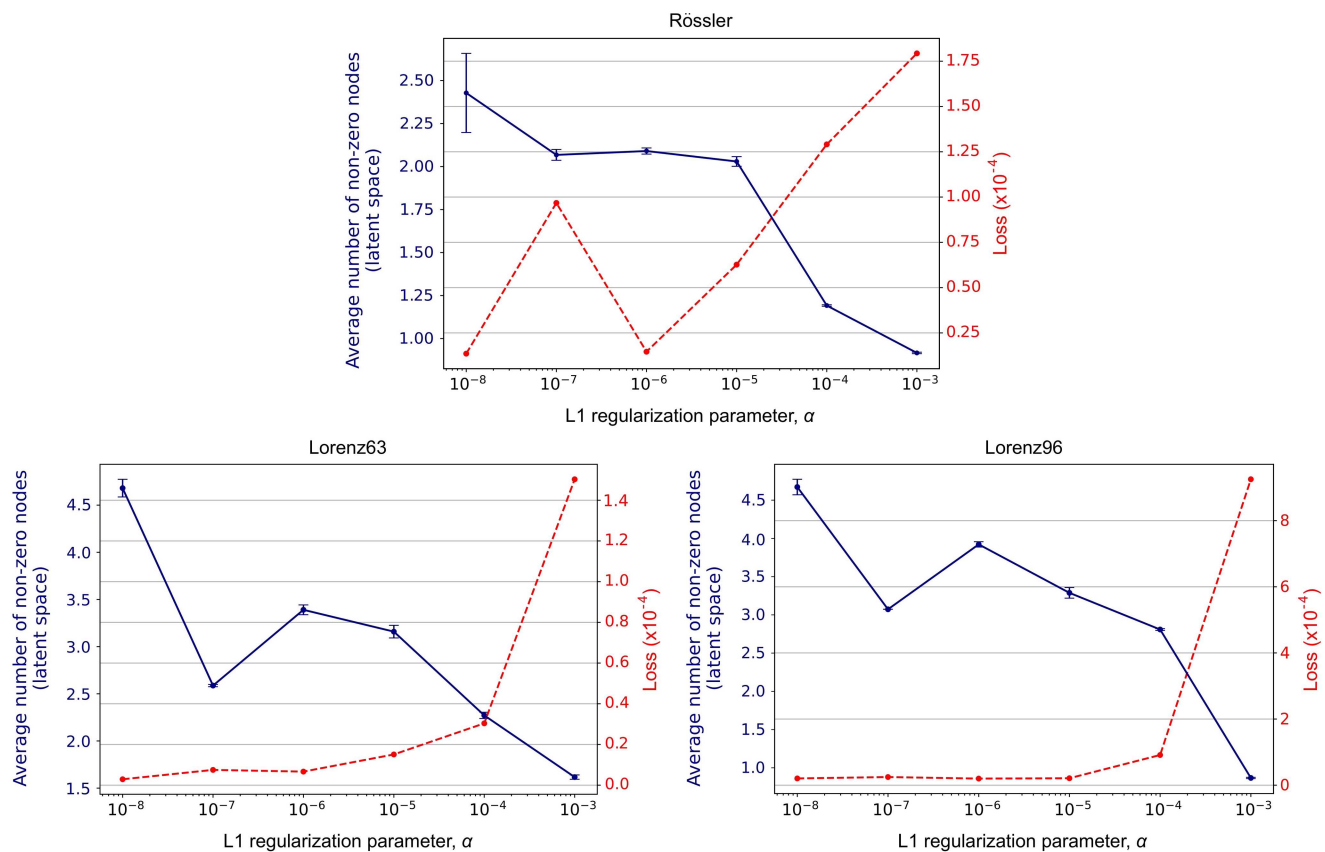


FIG. 3. The average number of non-zero nodes in latent space (navy blue solid line), including the standard deviation of it (error bars) and the corresponding value of the loss function (red dashed line) of the test data as a function of the L1 regularization parameter, α , for the three systems.

from each other will diverge exponentially if the attractor is chaotic. The exponential discrepancy is described by the largest Lyapunov exponent (LLE). This definition can be extended to the spectrum of Lyapunov exponents. The LLE of a system can be calculated using two approaches. The first approach uses a Jacobian matrix and can be applied when the mathematical model describing the evolution of the system is known. This method gives the Lyapunov spectrum. The second one can be used to estimate the LLE of a time series by applying the definition of the local divergence on a segment of a given trajectory. Estimating the LLE from a time series faces several numerical difficulties, like dealing with the noise in the data.^{10,12,13} The basic definition of the LLE, λ_1 , can be written as

$$d(t) \approx Ce^{\lambda_1 t}, \quad (6)$$

where $d(t)$ is an average distance at time, t , and C is a constant (usually normalized by the initial separation of neighbors). Here, we use the methodology proposed by Rosenstein *et al.*,¹² where we divide the time, t , into steps of Δt , and assume that at time, $t = i \times \Delta t$, the distance of the j th pair of nearest neighbors, $d_j(i)$, diverges approximately at a rate given by the LLE,

$$d_j(i) \approx C_j e^{\lambda_1 i \Delta t}, \quad (7)$$

where C_j is the initial distance of the selected pair of points. By taking the logarithm of Eq. (7), we have that

$$\ln d_j(i) \approx \ln C_j + \lambda_1 (i \Delta t). \quad (8)$$

Equation (8) represents a set of M (all the values of index j) approximately parallel lines that can be fitted using least squares to calculate an averaged value and its standard deviation of the LLE.

In this work, we compare the average value and its standard deviation of the LLE calculated using the original and the reconstructed trajectories. The reconstructed dynamics obtained by the autoencoder depend on the parameter in L1 regularization— α . We report these findings and the dependence of LLE on the embedding dimension window in Sec. IV.

IV. RESULTS

The basic results of this work consist of the evaluation of the effective size of the latent space of the chaotic autoencoder constructed for each of the three systems we considered. The size of the latent space depends on the regularization parameter α and the window size of the input sequence of the time series, W . Once the

TABLE II. The largest Lyapunov exponent and its standard deviation in parenthesis for the input sequence and for each one of the reconstructed ones as a function of the regularization parameter, α , for the three systems.

Sequence	α	Rössler			Lorenz63			Lorenz96		
		X	Y	Z	X	Y	Z	19	20	21
Input	...	1.73 (0.08)	1.72 (0.05)	1.74 (0.09)	1.17 (0.30)	1.17 (0.35)	1.14 (0.18)	3.02 (0.88)	3.03 (0.35)	3.06 (0.57)
	10^{-3}	1.76 (0.05)	1.73 (0.02)	1.77 (0.13)	1.16 (0.42)	1.13 (0.21)	1.14 (0.05)	2.99 (0.53)	2.63 (0.73)	2.49 (0.89)
	10^{-4}	1.74 (0.02)	1.73 (0.02)	1.73 (0.01)	1.14 (0.13)	1.12 (0.11)	1.14 (0.34)	3.08 (0.57)	2.79 (0.53)	3.03 (0.57)
	10^{-5}	1.77 (0.09)	1.73 (0.06)	1.74 (0.09)	1.14 (0.16)	1.17 (0.34)	1.13 (0.15)	2.88 (0.57)	2.83 (0.44)	3.01 (0.54)
Reconstructed	10^{-6}	1.78 (0.09)	1.73 (0.04)	1.78 (0.01)	1.14 (0.26)	1.17 (0.36)	1.14 (0.24)	2.88 (0.63)	3.01 (0.22)	2.90 (0.42)
	10^{-7}	1.76 (0.07)	1.74 (0.02)	1.78 (0.05)	1.15 (0.29)	1.17 (0.26)	1.13 (0.35)	2.97 (0.41)	2.83 (0.56)	3.04 (0.43)
	10^{-8}	1.73 (0.25)	1.73 (0.11)	1.79 (0.05)	1.17 (0.27)	1.17 (0.29)	1.14 (0.07)	2.87 (0.27)	3.04 (0.64)	3.01 (0.50)

autoencoder is trained, we also compare the largest Lyapunov value it produces.

A. Latent space size as a function of the regularization parameter, α

We apply the methods outlined above on the three chaotic systems, shown in Table I. In Fig. 3, we present the average number

of non-zero nodes at the latent space (nodes with output value larger than α) as well as the loss of the reconstructed sequence, with respect to the input sequence, as a function of the L1 regularization parameter, α .

We conclude that we find the same patterns for all studied systems, i.e., the loss function increases with the increment of the regularization parameter. Depending on which level of accuracy could be needed, the number of active nodes might change. In

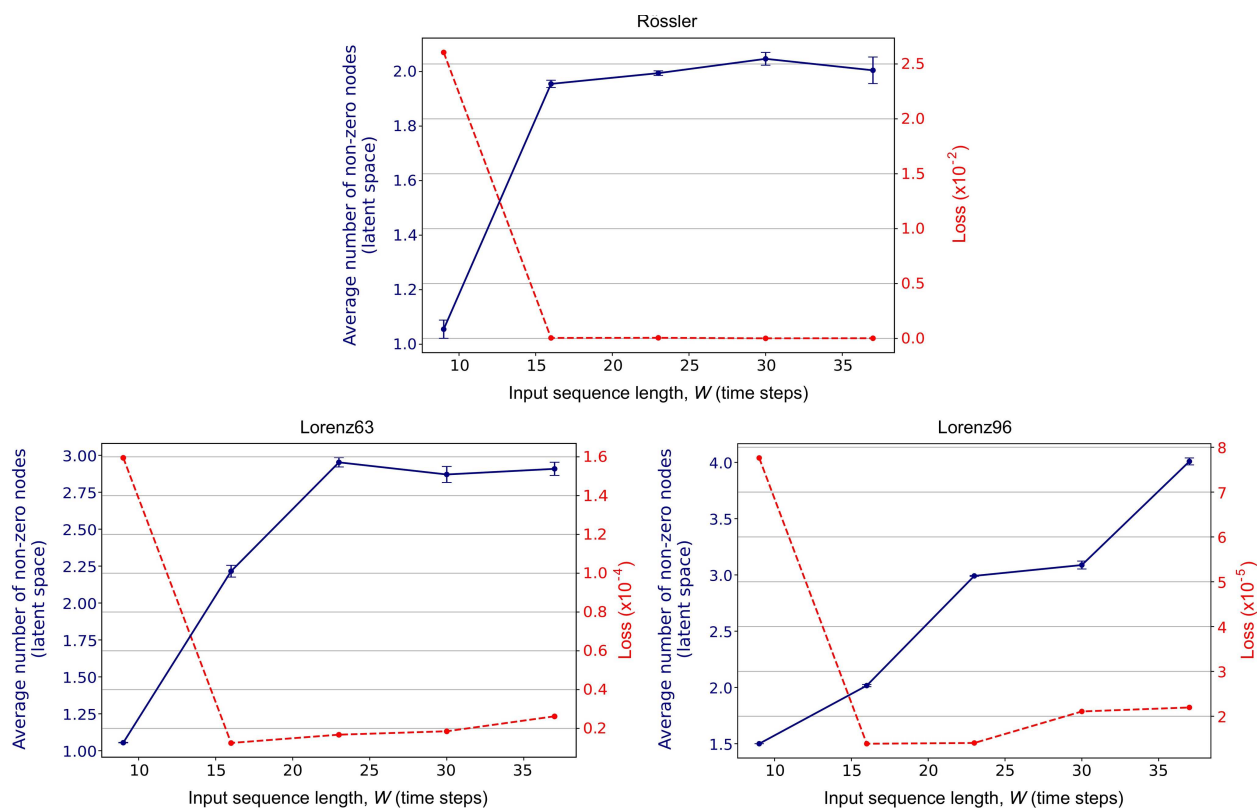


FIG. 4. The average number of non-zero nodes in latent space (navy blue solid line), including the standard deviation of it (error bars) and the corresponding value of the loss function (red dashed line) of the test data as a function of the size of the input sequence, W , for the three chaotic systems of Table I.

particular, we observe that for the Rössler system, the autoencoder needs two nodes in the latent space to reconstruct the input sequence with error less than 10^{-4} , while three to four nodes are needed in the case of the Lorenz systems.

To further demonstrate the efficiency of the reconstructed trajectories by the autoencoder model, we calculate the largest Lyapunov exponent for the input and the reconstructed sequences of each one of the regularization parameter values. We calculate the LLE by creating two trajectories of each attractor. We choose a starting point for one of them and then we displace it by 10^{-7} for the second trajectory. We create the two trajectories by solving the equations of each attractor and then we use the trained autoencoder of the corresponded system to reconstruct the trajectories. In Table II, we present the LLE and its standard deviation in parenthesis for the input sequence, and the reconstructed sequence of each one of the values of the regularization parameter, α . The results indicate that the reconstructed sequences created by models with α smaller or equal to 10^{-4} have the same characteristics as the sequences obtained by solving the deterministic system of equations. The error in the average value of LLE is less than $\pm 2\%$.

The obtained agreement between the system and chaotic autoencoder LLE is quite promising since it shows that the latter acts as an effectively linear system driven by a chaotic signal. It, thus, demonstrates further that the chaotic autoencoder is a faithful reconstruction of the original system in the specific parameter regime.

B. Latent space dimension as a function of the size of the input sequence, W

It is important to know how the size of the information passed to the autoencoder (size of input sequence) affects the performance of the model. In Fig. 4, we present the average number of non-zero nodes in the latent space that were used to reconstruct the input sequence as a function of the size of the input sequence, W . We fix the regularization parameter α to 10^{-5} and range the size of the input sequence, W from 9 to 37 time steps using steps of 7 time steps.

In the case of the Rössler system, the autoencoder needs two non-zeros nodes in the latent space to reconstruct a sequence of more than 16 time steps. Three non-zero nodes are needed in the Lorenz63 model to reconstruct a sequence of 23 time steps. We observe again that the Lorenz96 system is more complex than the other two, and the adequate number of nodes to represent it depends

strongly on the desired accuracy. In Table III, we present the LLE and its standard deviation in parenthesis for the input sequence, and the reconstructed sequence of each one of the values of the size of the input sequence, W . The error in the average value of LLE is found to be less than 5% in all cases.

V. CONCLUSIONS

In this work, we attempted to connect chaotic dynamical systems with neural network-based autoencoders and find the minimal set of NN variables that may reproduce these dynamical systems. The construction of chaotic autoencoders is performed through deep learning networks, and in the process of their use, an information bottleneck is formed. The latter is determined exclusively by the dimension of the latent space. We aimed to find the minimal dimension of the latent space that faithfully reproduces the chaotic systems we studied. This dimension depends on the required accuracy at the output. Furthermore, it also depends on the window size we choose in order to perform the attractor reconstruction. The analysis of the latent space dimensionality as a function of the regularization parameter and the input sequence size gives compatible numbers and this feature adds to the consistency of the picture. In addition, the resulting maximal Lyapunov exponent of the chaotic autoencoder is, respectively, very close to each of the chaotic systems we analyzed. This is a significant outcome since it shows that the chaotic autoencoder can represent faithfully complex, irregular dynamics.

An important issue that opens up from the present work is the connection of the dimensionality of the latent space to that of the embedding dimension of the time series. Intuitively, one might think that the latent space is in some sense the minimal space required to represent a chaotic dynamical system. As a result, the latent space dimensions should be smaller or at least equal to the embedding dimension since the latter gives the framework space for analyzing the chaotic series. We observe that this feature is true in the examples we addressed since we find latent space dimension smaller or equal to the number of differential equations that generate respective data. An additional appealing outcome of the present analysis is that the largest Lyapunov exponent that is connected with the autoencoder system appears to be very close to the one determined directly from the dynamical system. Clearly, these two exponents are evaluated differently since the chaotic autoencoder cannot by construction

TABLE III. The largest Lyapunov exponent and its standard deviation in parenthesis for the input sequence and for each of the reconstructed ones as a function of the size of the input sequence, W , for the three chaotic systems of Table I.

Sequence	W	Rössler			Lorenz63			Lorenz96		
		X	Y	Z	X	Y	Z	19	20	21
Input	...	1.73 (0.08)	1.72 (0.05)	1.74 (0.09)	1.17 (0.30)	1.17 (0.35)	1.14 (0.18)	3.02 (0.88)	3.03 (0.35)	3.06 (0.57)
	9	1.77 (0.02)	1.75 (0.04)	1.70 (0.19)	1.15 (0.37)	1.17 (0.34)	1.15 (0.16)	2.92 (0.40)	2.92 (0.64)	3.95 (0.51)
	16	1.73 (0.21)	1.72 (0.34)	1.80 (0.12)	1.13 (0.18)	1.13 (0.34)	1.13 (0.40)	2.95 (0.64)	3.05 (0.57)	3.08 (0.21)
Reconstructed	23	1.73 (0.21)	1.76 (0.08)	1.67 (0.44)	1.14 (0.40)	1.15 (0.32)	1.15 (0.51)	2.98 (0.39)	2.92 (0.59)	3.08 (0.38)
	30	1.76 (0.14)	1.73 (0.05)	1.73 (0.85)	1.14 (0.29)	1.18 (0.13)	1.18 (0.44)	2.94 (0.52)	2.97 (0.76)	2.92 (0.41)
	37	1.74 (0.16)	1.74 (0.13)	1.82 (0.79)	1.14 (0.18)	1.16 (0.44)	1.17 (0.41)	2.96 (0.35)	3.03 (0.36)	3.02 (0.46)

generate its dynamics and depends on the input time series. The Lyapunov exponent, in this case, is obtained by observing the proximity of two nearby chaotic series with a very small difference in their initial conditions. However, the fact that the autoencoder can generate similar divergence in the difference of the two series as that of the original system shows that it provides a faithful representation of the original system. In other words, we find that the chaotic autoencoder provides a faithful representation of time series segments and the more intricate internal dynamics of the chaotic system.

The present work opens up an information window in dynamical systems. It demonstrates that AI methods, when applied properly, may give information on complex dynamical systems and their various reduced representations. This information is both quantitative and provides additional representations of dynamical systems. In some sense, it also shows that neural networks are ubiquitous and may capture accurately a high degree of information contained in complex dynamical systems. This result treated a bit more philosophically may be seen as a rather optimistic one for us since it demonstrates that human neural networks are indeed capable of capturing the complexity of the outside world.

ACKNOWLEDGMENTS

We acknowledge the co-financing of this research by the European Union and Greek national funds through the Operational Program Crete 2020-2024, under the call “Partnerships of Companies with Institutions for Research and Transfer of Knowledge in the Thematic Priorities of RIS3Crete,” with project title “Analyzing urban dynamics through monitoring the city magnetic environment” (Project KPHP1—No. 0029067) and also by the Ministry of Science and Higher Education of the Russian Federation in the framework of Increase Competitiveness Program of NUST “MISiS” (No. K2-2019-010), implemented by a governmental decree dated March 16, 2013, N 211.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

- ¹V. S. Anishchenko, V. Astakhov, A. Neiman, T. Vadivasova, and L. S.-Geier, *Nonlinear Dynamics of Chaotic and Stochastic Systems*, 2nd ed. (Springer-Verlag, Berlin, 2007).
- ²H. Abarbanel, *Analysis of Observed Chaotic Data* (Springer-Verlag, New York, 1996).
- ³H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis* (Cambridge University Press, 2004), Vol. 7.
- ⁴E. Bradley and H. Kantz, “Nonlinear time-series analysis revisited,” *Chaos* **25**, 097610 (2015).
- ⁵H. Abarbanel, R. Brown, J. J. Sidorowich, and L. S. Tsimring, *Rev. Mod. Phys.* **65**, 1331 (1993).
- ⁶E. R. Deyle and G. Sugihara, “Generalized theorems for nonlinear state space reconstruction,” *PLoS One* **6**(3), e18295 (2011).
- ⁷P. Verdes, P. Granitto, H. Navone, and H. Ceccatto, “Forecasting chaotic time series: Global vs. local methods,” *Novel Intell. Automat. Control Syst.* **1**, 129–145 (1998).

- ⁸I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning* (MIT Press, 2016).
- ⁹M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science* **349**(6245), 255 (2015).
- ¹⁰A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano, “Determining Lyapunov exponents from a time series,” *Physica D* **16**, 285 (1985).
- ¹¹M. Sano and Y. Sawada, “Measurement of the Lyapunov spectrum from a chaotic time series,” *Phys. Rev. Lett.* **55**, 1082 (1985).
- ¹²M. T. Rosenstein, J. J. Collins, and C. J. De Luca, “A practical method for calculating largest Lyapunov exponents,” *Physica D* **65**(1-2), 117–134 (1993).
- ¹³J. P. Eckmann, S. O. Kamphorst, D. Ruelle, and S. Ciliberto, “Lyapunov exponents from time series,” *Phys. Rev. A* **34**, 4971–4979 (1986).
- ¹⁴U. Parlitz, “Identification of true and spurious Lyapunov exponents from time series,” *Int. J. Bifurcation Chaos* **01**, 155 (1992).
- ¹⁵M. Nielsen, *Neural Networks and Deep Learning* (Determination Press, San Francisco, 2015).
- ¹⁶Z. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott, “Reservoir observers: Model-free inference of unmeasured variables in chaotic systems,” *Chaos* **27**, 041102 (2017).
- ¹⁷R. Budhiraja, M. Kumar, M. K. Das, A. S. Bafila, and S. Singh, “A reservoir computing approach for forecasting and regenerating both dynamical and time-delay controlled financial system behavior,” *PLoS One* **16**(2), e0246737 (2021).
- ¹⁸P. R. Vlachas, W. Byeon, Z. Y. Wan, T. P. Sapsis, and P. Koumoutsakos, “Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks,” *Proc. R. Soc. A* **474**, 2213 (2018).
- ¹⁹G. Neofotistos, M. Mattheakis, G. D. Barmparis, J. Hizanidis, G. P. Tsimonis, and E. Kaxiras, “Machine learning with observers predicts complex spatiotemporal behavior,” *Front. Phys.* **7**, 24 (2019).
- ²⁰S. L. Brunton, J. L. Proctor, and J. N. Kutz, “Discovering governing equations from data by sparse identification of nonlinear dynamical systems,” *Proc. Natl. Acad. Sci. U.S.A.* **113**, 3932–3937 (2016).
- ²¹F. Dercole, M. Sangiorgio, and Y. Schmirander, “An empirical assessment of the universality of ANNs to predict oscillatory time series,” *IFAC-Papers Online* **53**(2), 1255–1260 (2020).
- ²²B. Lusch, J. N. Kutz, and S. L. Brunton, “Deep learning for universal linear embeddings of nonlinear dynamics,” *Nat. Commun.* **9**, 4950 (2018).
- ²³W. H. Lopez Pinaya, S. Vieira, R. Garcia-Dia, and A. Mechelli, “Autoencoders,” in *Machine Learning: Methods and Applications to Brain Disorders* (Elsevier, 2020), pp. 193–208.
- ²⁴M. Kramer, “Nonlinear principal component analysis using autoassociative neural networks,” *AIChE J.* **37**(2), 233–243 (1991).
- ²⁵T. Sauer, J. A. Yorke, and M. Casdagli, “Embedology,” *J. Stat. Phys.* **65**(3), 579–616 (1991).
- ²⁶R. Hilborn, *Chaos and Nonlinear Dynamics: An Introduction for Scientists and Engineers* (Oxford University Press, 2000).
- ²⁷E. N. Lorenz, “The local structure of a chaotic attractor in four dimensions,” *Physica D* **13**, 90–104 (1984).
- ²⁸D. S. Broomhead and G. P. King, “Extracting qualitative dynamics from experimental data,” *Physica D* **20**(2), 217–236 (1986).
- ²⁹F. Takens, *Dynamical Systems and Turbulence*, Lecture Notes in Mathematics, edited by D. A. Rand and L. S. Young (Springer-Verlag, Berlin, 1981), pp. 366–381.
- ³⁰N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw, “Geometry from a time series,” *Phys. Rev. Lett.* **45**(9), 712–716 (1980).
- ³¹M. T. Rosenstein, J. J. Collins, and C. J. De Luca, “Reconstruction expansion as a geometry-based framework for choosing proper delay times,” *Physica D* **73**(1–2), 82–98 (1994).
- ³²D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv:1412.6980v9* (2017).
- ³³X. Glorot and Y. Bengio, in *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics* (PMLR, 2010), Vol. 9, pp. 249–256.
- ³⁴M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard et al., “Tensorflow: A system for large-scale machine learning,” in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI’16)* (USENIX Association, 2016), p. 265.
- ³⁵F. Chollet et al., Keras (2015), see <https://keras.io>.