CCFE

CULHAM CENTRE FOR FUSION ENERGY

UK Atomic Energy Authority

UKAEA-CCFE-PR(19)66

N. Boulle, V. Dallas, Y. Nakatsukasa, D. Samaddar

Classification of chaotic time series with deep learning

Classification of chaotic time series with deep learning

N. Boulle, V. Dallas, Y. Nakatsukasa, D. Samaddar

This is a preprint of a paper submitted for publication in
Physica D

Classification of chaotic time series with deep learning

Nicolas Boullé[a,*], Vassilios Dallas[a], Yuji Nakatsukasa[a], D. Samaddar[b]

[a] *Mathematical Institute, University of Oxford, Oxford, OX2 6GG, UK*
[b] *United Kingdom Atomic Energy Authority, Culham Centre for Fusion Energy, Culham Science Centre, Abingdon, Oxon, OX14 3DB, UK*

Abstract

We use deep neural networks to classify time series generated by discrete and continuous dynamical systems based on their chaotic behavior. Our approach to circumvent the lack of precise models for some of the most challenging real-life applications is to train different neural networks on a data set from a dynamical system with a basic or low-dimensional phase space and then use these networks to classify time series of a dynamical system with more intricate or high-dimensional phase space. We illustrate this extrapolation approach using the logistic map, the sine-circle map, the Lorenz system, and the Kuramoto–Sivashinsky equation. We observe that the proposed convolutional neural network with large kernel size outperforms state-of-the-art neural networks for time series classification and is able to classify time series as chaotic or non-chaotic with high accuracy.

Keywords: Dynamical systems, Chaos, Deep learning, Time series, Classification

1. Introduction

Data and in particular time series are generated from numerous observations and experiments across different scientific fields such as atmospheric and oceanic sciences for climate predictions, nuclear fusion for control and safety, biology and medicine for diagnosis. Fourier transforms, radial basis functions approximation and standard numerical techniques have been extensively applied to perform short and long term predictions of chaotic time series [1, 2, 3, 4]. On the other hand, the spectacular success of machine learning and deep learning techniques to image classification [5, 6], which have recently surpassed human-level performance on the ImageNet data set [7], has inspired the development of neural network techniques for time series forecasting [8, 9] and classification [10]. Recently, deep learning approaches have been used to solve partial differential equations in high dimensions [11, 12, 13] and identify hidden physics models from experimental data [14, 15, 16, 17].

The size of the data sets is often large and analysing these time series represents a huge computational challenge and interest nowadays. For some of the most challenging real-life applications a precise dynamical system is unknown, which makes the identification of the different dynamical regimes impossible. On that spirit, machine learning has been recently employed by Pathak et al. [18, 19] to perform model-free predictions of chaotic dynamical systems. Moreover, deep learning requires a large data set to adequately train the artificial neural network, which might not be available in some cases due to the infinite dimensional phase space of the system or experimental constraints.

In this paper, we address the aforementioned challenges by considering the problem of classifying time series generated by discrete and continuous dynamical systems according to their potential chaotic behavior. The problem can be stated as follows: given a time series generated by a dynamical system, can we determine whether the time series has a chaotic or non-chaotic behavior ? Contrary to standard machine learning techniques, we choose to train the neural network on a different set than the testing set of interest in order to assess the ability of the machine learning algorithms to extrapolate

*Corresponding author
Email addresses: boulle@maths.ox.ac.uk (Nicolas Boullé), vassilios.dallas@gmail.com (Vassilios Dallas), nakatsukasa@maths.ox.ac.uk (Yuji Nakatsukasa), debasmita.samaddar@ukaea.uk (D. Samaddar)

the classification of time series of a dynamical system with a basic or low-dimensional phase space to a more intricate or high-dimensional one. The main challenge is to learn the chaotic features of the training set, whose chaotic behavior can be determined a priori using measures from dynamical systems theory, without overfitting, and generalise on a second data set, which behaves differently due to the different dynamical behavior of the system.

The paper is organised as follows. We briefly describe five different neural networks architectures for time series classification in Section 2. Then, in Section 3, we classify signals generated by discrete dynamical systems and compare the accuracy of the neural networks. Finally, Section 4 consists of the classification of time series generated by the Lorenz system and the Kuramoto–Sivashinsky equation.

2. Neural networks for time series classification

Time series classification is one of the most challenging problems in machine learning with a wide range of applications in human activity recognition [20], acoustic scene classification [21], and cybersecurity [22]. Fawaz et al. provide a review of the techniques used for time series classification and compare different neural networks [23].

In this section, we describe five different architectures that we have considered for classifying time series generated by discrete and continuous dynamical systems. Three of these methods have been studied by Wang, Yan, and Oates [10] and are discussed in Sections 2.3 to 2.5. In the following subsections, we assume that the input of the neural network is a univariate time series X of size T (in practice we take T to be one thousand). Each of the time series is assigned a class label that we want to recover using the different neural networks: Class 1 corresponds to a non-chaotic time series while Class 2 corresponds to a chaotic time series.

2.1. Shallow neural network

The first type of networks considered in this section is shallow neural networks (ShallowNet), which are simple and efficient networks for fitting functions and perform pattern recognition. These networks differ from deep neural network since they usually contain only one or two hidden layers. We use MATLAB's `patternnet` command from the Deep Learning Toolbox to define a network with one hidden layer, containing one hundred neurons, with the sigmoid as activation function. The network is trained with the Scaled Conjugate Gradient algorithm [24].

2.2. Convolutional neural network

Convolutional neural networks (ConvNet) have first been introduced to perform handwritten digit [25] and have been successfully applied to images and time series [6, 26]. We apply a similar technique to classify one dimensional time series and consider a network composed of two convolutional layers of five features and kernel size of a hundred, followed by a rectified linear unit (ReLU), a maximum pooling and two fully connected layers of respective size one hundred and two. The ReLU activation function is chosen because it is easy to optimise due to its piecewise linearity [27, Chap. 6]. Moreover, a dropping out unit (dropout) is added after the maximum pooling layer to improve the generalisation ability of the network [28]. The architecture of the network is shown in Figure 1.

Traditional implementations of convolutional neural networks usually consider a higher number of features and a much smaller kernel size [10, 23]. A highlight of the neural network considered in this section is the large kernel size of the convolutional layer (one hundred), which we determined necessary by experimentation to overcome the overfitting issue on the data sets discussed in Sections 3 and 4. Moreover, we also choose to use a reduced number of features for computational purposes.

2.3. Multi layer perceptrons

Multi layer perceptrons (MLP) are standard deep neural network architectures in the field of machine learning and essentially consist of fully connected layers separated by a nonlinear activation function. Wang, Weizhong and Oates [10] use a structure of three hidden layers of five hundred neurons, with dropout at each layer followed by a rectified linear unit, to perform time series classification (a Python implementation using TensorFlow is available in [29]).

2.4. Fully convolutional neural network

As explained in Section 2.2, convolutional neural networks have shown good efficiency on image classification problems. The fully convolutional neural network (FCN) architecture considered in [10] is a succession of three convolutional blocks, followed

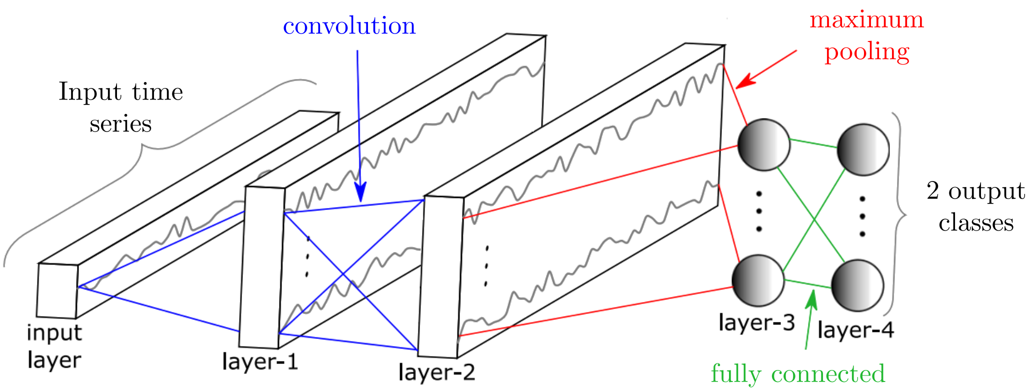

Figure 1: A convolutional neural network architecture for time series classification. Figure adapted from [23].

by a global averaged pooling layer [30]. The blocks are composed of a convolution layer of kernel size 8, 5, and 3, a batch normalization layer [31] and a ReLu activation layer. The fully convolutional neural network studied by [10] is implemented in [29].

2.5. Residual network

The last network considered by Wang, Weizhong and Oates [10] in the context of time series classification is a residual network (ResNet). Residual networks are examples of very deep neural network and are designed by stacking the convolutional blocks arising in the FCN (see Section 2.4). Then, the ResNet is created by assembling three blocks of the FCN to generate a residual block. Three residual blocks are then stacked and followed by a global average pooling layer and a softmax layer to output the classification of the different input time series. Reference [29] provides a practical Python implementation.

3. Discrete dynamical systems

In this section we consider two discrete dynamical systems called the logistic map and the sine-circle map. The first one is the logistic map, popularized by Robert May [32], which is defined by the sequence

$$x_{n+1} = \mu x_n(1 - x_n), \quad x_0 = 0.5, \tag{1}$$

where μ is the bifurcation parameter varying between zero and four. This system exhibits periodic or chaotic behavior depending on the value of μ. Periodic and chaotic signals of the logistic map are plotted in Figure 2.

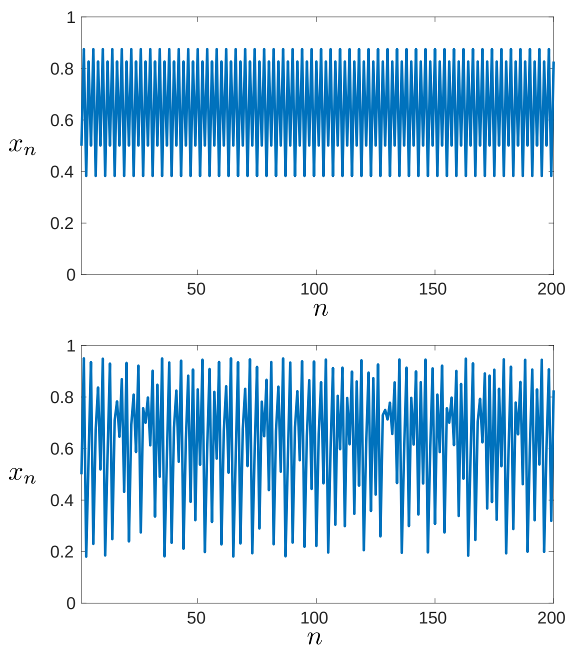

Figure 2: A periodic (top) and a chaotic (bottom) signal of the logistic map of size two hundred with $\mu = 3.5$ and $\mu = 3.8$, respectively.

The bifurcation diagram showing the orbits of the logistic map is represented in Figure 3 (top). The behavior of the attractors for different parameters μ has been extensively studied [33, Chap. 10] and a highlight is the period-doubling cascade happening for $\mu \in [0, 3.54409]$.

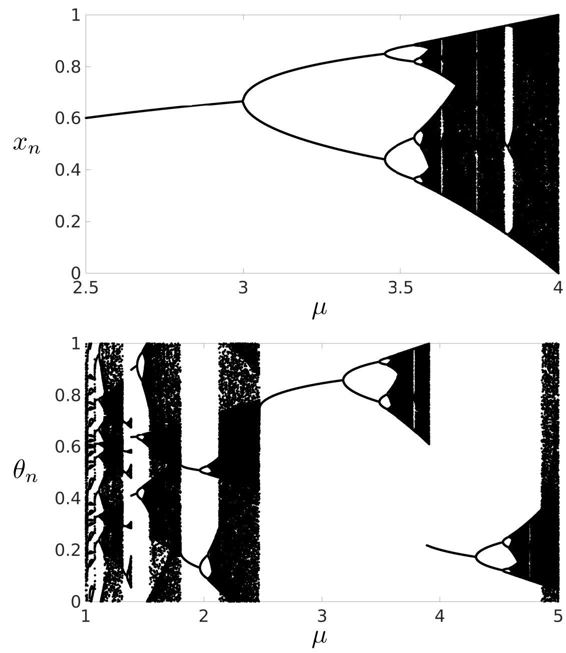

Figure 3: Bifurcation diagrams of the logistic map (top) and the sine-circle map (bottom).

The second dynamical system considered in this section is the sine-circle map [34, Chap. 6], which is sometimes referred to as the circle map. It takes the form of the following nonlinear map

$$\theta_{n+1} = \theta_n + \Omega - \frac{\mu}{2\pi} \sin(2\pi\theta_n) \mod [1], \quad \theta_0 = 0.5, \tag{2}$$

where $\Omega = 0.606661$ and $\mu \in [0, 5]$ is the parameter that measures the strength of the nonlinearity. Similarly to the logistic map, iterating Equation (2) leads to periodic or chaotic signals depending on the bifurcation parameter μ chosen. Figure 4 illustrates two signals with different behaviors, generated using a bifurcation parameter of $\mu = 2.1$ (top) and $\mu = 2.3$ (bottom). The bottom panel of Figure 3 shows the bifurcation diagram of the sine-circle map.

We now want to classify signals generated by the logistic and sine-circle maps according to their chaotic and non-chaotic behavior. Our main goal, and challenge, is to find a neural network that is

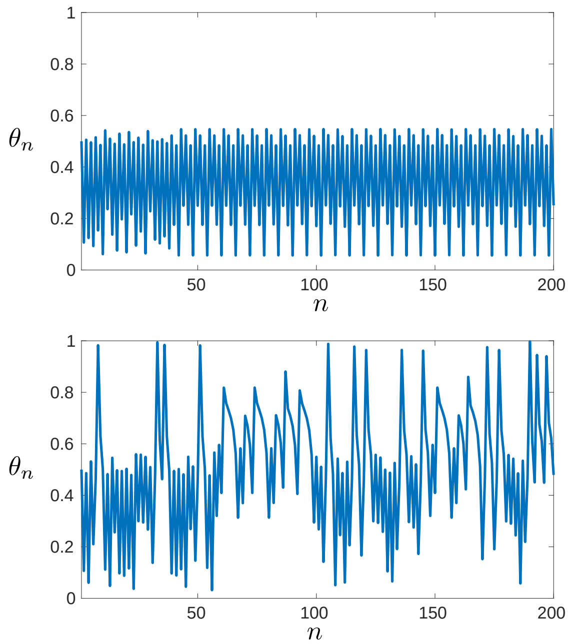

Figure 4: A periodic (top) and a chaotic (bottom) signal of the sine-circle map of size two hundred with $\mu = 2.1$ and $\mu = 2.3$, respectively.

able to learn the features characterising chaotic signals of the logistic map and generalise on signals generated by the sine-circle map. To do this, we first generate two data sets by computing signals of length one thousand of the logistic and sine-circle maps for five thousand different values of the parameter μ. To construct our training set we employ a sampling technique that satisfies a uniform distribution across the two thirds of the logistic map data set, which we use as a training set for our machine learning algorithms. The other one third of the data set is then used as a validation set to check that the training has been successful. Then, the five neural networks described in Section 2 are used to classify the time series of the sine-circle data set.

The classification of the time series is done using two measures from dynamical systems theory. The first measure is the Lyapunov exponent which is defined as

$$\lambda = \lim_{n \to +\infty} \frac{1}{n} \sum_{i=0}^{n-1} \log |f'(x_i)| \tag{3}$$

for a discrete dynamical system $x_{n+1} = f(x_n)$ and expresses the exponential separation, viz. $d(t) = d_0 e^{\lambda t}$, of two nearby trajectories originally separated by distance $d_0 = \epsilon \ll 1$ at time $t = 0$. The

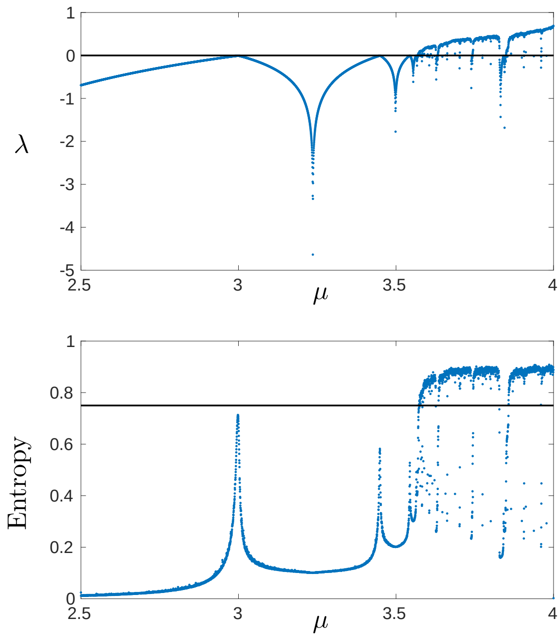

Figure 5: Lyapunov exponents (left) and Shannon entropy (right) of the logistic map. A time series is chaotic if its Lyapunov exponent is greater than zero and its entropy greater than 0.75. The horizontal black lines in the plots indicate these thresholds.

second measure is the Shannon entropy which uses the probability distribution function of a trajectory to quantify a range of accessible states for a dynamical system and relates to the potential topological transitivity of the system [34, Chap. 9]. Hence, we expect a chaotic system to have well distributed trajectories in space compared to a periodic one and the aim is then to count the number of accessible states for the system. Thus, we define the Shannon entropy of a time series x_n to be

$$S_N = -\frac{1}{\log(N)} \sum_r p_r \log(p_r), \tag{4}$$

where p_r is the probability to be on the state r reached by the system with $p_r = \frac{1}{N}\#\{x_i = r | 1 \leq i \leq N\}$ and N is the number of sample points. Here, the entropy S_N has been normalised to lie in $[0, 1]$ so that the entropy of a constant signal $S_N \to 0$ while the entropy of a chaotic time series $S_N \to 1$.

We classify a given signal as chaotic when its Lyapunov exponent is strictly positive and its entropy is greater than a given threshold, experimentally set at 0.75 (see also Figure 5), and non-chaotic otherwise. It is crucial to classify the training data set accurately in order to reduce misclassifications on the testing set. On that note, by using the Shannon entropy in addition to the Lyapunov exponent as a measure of chaos we gained an incremental improvement in accuracy. This is because the Lyapunov exponent was misclassifying some quasi-periodic signals as chaotic.

The Lyapunov exponent and the Shannon entropy of the logistic map as a function of the bifurcation parameter μ are illustrated in Figure 5. In real applications, computing these quantities over the whole range of parameters and in some cases without knowing the expression of the underlying dynamical system can be unfeasible or computationally expensive, which justifies the approach of using a machine learning algorithm to perform the classification automatically.

The average classification accuracy of the neural networks ShallowNet, ConvNet, MLP, FCN, and ResNet is reported in Table 1. While the ShallowNet and ResNet architectures outperforms the convolutional neural network ConvNet on the logistic map data set with a score above 99%, they are only able to classify correctly signals from the sine-circle dynamical system with an accuracy less than 65%. The convolutional network however seems to override overfitting issues on the training set by capturing the main features of chaotic and periodic signals and gets an average classification score of 83.5%. It is of interest to notice that the shallow neural network reaches an accuracy greater than state-of-the-art time series classification networks on the sine-circle data set despite its simplicity. Improving the accuracy of ConvNet on the sine-circle map might be challenging since this dynamical system could lead to behaviors of signals that are not available in the training set of the logistic map (e.g. the regime $\mu \in [1, 1.3]$ in Figure 3 (bottom)).

4. Continuous dynamical systems

We now consider continuous dynamical systems of ordinary and partial differential equations that exhibit temporal and spatiotemporal chaos, respectively. The aim here is to determine whether a neural network trained on a low dimensional dynamical system is able to generalise and classify time series generated by a higher dimensional dynamical system. We will first consider the Lorenz system since it is one of the most typical continuous dynamical systems with a chaotic behavior which has been widely studied in the twentieth century [35].

Table 1: Classification score on the logistic and sine-circle maps data sets. The neural networks are trained on logistic signals and the accuracy is averaged over five training cycles.

Networks	ShallowNet	ConvNet	MLP	FCN	ResNet
Logistic	99.5	93.6	83.4	95.3	99.0
Sine-circle	64.9	83.5	60.2	54.0	54.6

4.1. Lorenz system

The Lorenz system [36] consists of the following three ordinary differential equations:

$$\dot{x} = \sigma(y - x), \tag{5a}$$
$$\dot{y} = x(\rho - z) - y, \tag{5b}$$
$$\dot{z} = xy - \beta z. \tag{5c}$$

The parameters $\sigma = 10$, $\beta = 8/3$, and $\rho \in [0, 250]$ yield convergent, periodic, and chaotic solutions. We numerically solve Equation (5) using MATLAB's function `ode45` with $[x, y, z] = [1, 1, 1]$ as initial condition. Integrating the equations for $t \in [0, 100]$ we obtain time series for $x(t)$, $y(t)$, and $z(t)$ of length one thousand, and we carry out this operation for five thousand values of the bifurcation parameter ρ in the range $[0, 250]$.

The time series $x(t)$, $y(t)$, and $z(t)$ are normalized by the linear transformation $x(t) \mapsto (x(t) - m)/(M - m)$, where M and m are respectively the maximum and minimum of the time series, such that their range are in the interval $[0, 1]$ (see time series in Figure 6 for $\rho = 70$). Figure 7 depicts four time series of the variable $x(t)$ generated by numerically solving Equation (5) for $\rho = 15$, 28, 160, and 180.

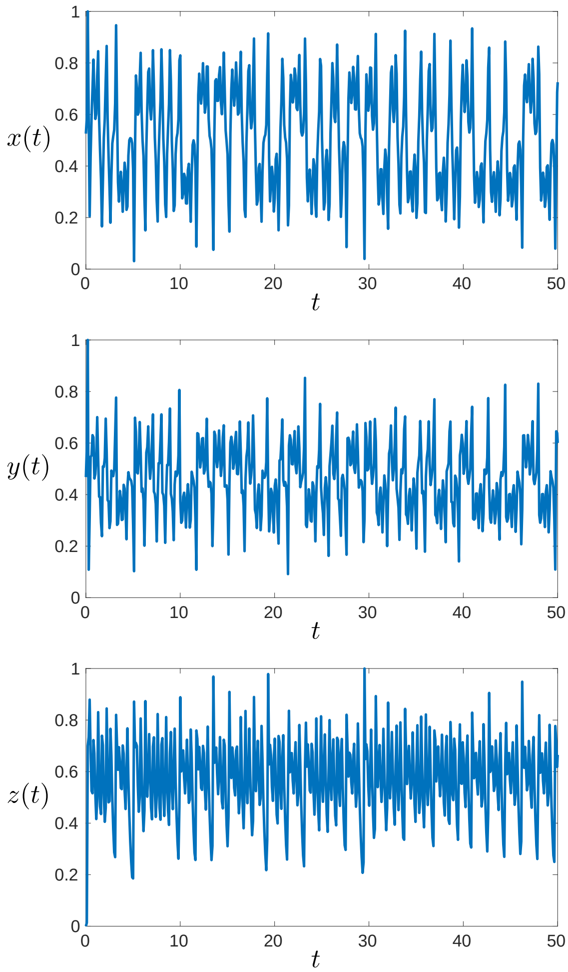

Figure 6: Normalized time series of the x, y, and z components of the Lorenz system with bifurcation parameter $\rho = 70$.

We classify the time series of the Lorenz system as chaotic or non-chaotic according to the sign of the Lyapunov exponent at the corresponding regimes of the bifurcation parameter ρ in order to generate training and testing data sets for the neural networks. Here, we compute the Lyapunov exponents for the time series of the variable $x(t)$ starting from some initial condition $x(0)$ as follows

$$\lambda = \lim_{t \to +\infty} \lim_{\epsilon \to 0} \frac{1}{t} \log \left(\frac{|x(t) - x_\epsilon(t)|}{\epsilon} \right), \tag{6}$$

where $|x(0) - x_\epsilon(0)| < \epsilon \ll 1$. Figure 8 shows the Lyapunov exponents of the variable $x(t)$, which determine the classification of the testing set of time series given to the neural networks described in Section 2. For example, the chaotic time series plotted in Figure 7 (d) corresponds to a bifurcation parameter of $\rho = 180$ and has a strictly positive Lyapunov

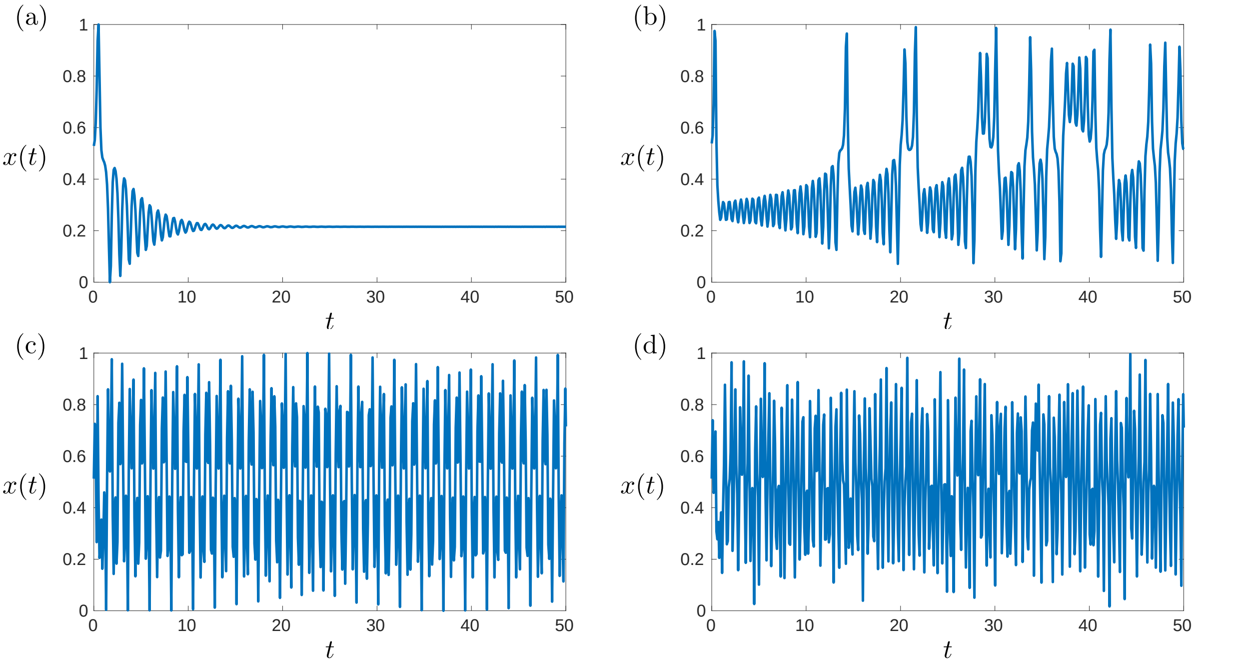

Figure 7: Normalized time series of the x component of the Lorenz system with bifurcation parameter $\rho = 15$ (a), 28 (b), 160 (c), and 180 (d).

exponent as shown in Figure 8. For continuous dynamical systems the Shannon entropy did not appear to be a precise measure of chaotic behavior, so we do not consider it in this case.

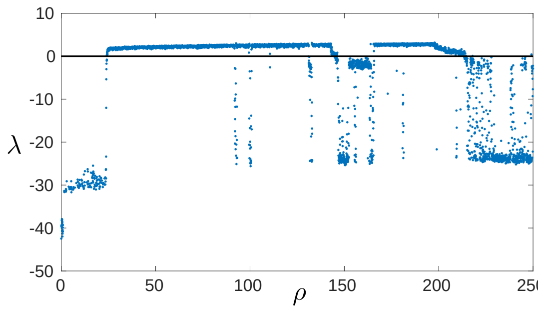

Figure 8: Lyapunov exponents of the $x(t)$ component of the Lorenz system for $\sigma = 10$, $\beta = 8/3$, and $\rho \in [0, 250]$. A positive Lyapunov exponent (points above the horizontal black line) indicates a chaotic solution to the Lorenz equations.

The different neural networks are trained on time series of the x component of the Lorenz system and tested on the y and z components. Similarly to the logistic map data set (see Section 3), the x component set is divided in the following way: two thirds for training and one third for validation. We then compare the classification accuracy of the networks described in Section 2 on the two data sets. The results are presented in Table 2.

The convolutional neural network ConvNet (see Section 2.2) outperforms the other networks on all the testing sets composed by time series of the x, y, and z components of the Lorenz system. In particular, it is able to generalise well on the z component by determining whether a given time series is chaotic or not correctly with an average accuracy of 79.4%. The other neural networks seem to overfit the training set and fail to classify time series of the z component correctly. Note that the y component of the Lorenz system is highly correlated with the x component, unlike the z component (see Figure 6), which explains the relative good classification accuracy (above 75%) of all the neural networks on the y component.

4.2. Kuramoto–Sivashinsky equation

In this section, we consider the Kuramoto–Sivashinsky (KS) equation, which is an example of a fourth-order nonlinear partial differential equation, which exhibits spatiotemporal chaos. This equation was originally derived by Kuramoto [37, 38, 39] and Sivashinsky [40, 41, 42] to model instabilities in laminar flame fronts and arises in a wide range of physical problems such as plasma physics [39], flame propagation [40], or free surface

Table 2: Classification score on the Lorenz system. The network is trained on x component of the Lorenz system and the accuracy is averaged over five training cycles.

Networks	ShallowNet	ConvNet	MLP	FCN	ResNet
Lorenz X	98.5	97.9	90.2	80.3	97.8
Lorenz Y	75.9	93.0	75.5	74.6	85.7
Lorenz Z	58.2	79.4	54.9	65.5	70.7

film flows [43, 44, 45]. In particular, we study the Kuramoto–Sivashinsky system normalized to the interval $[0, 2\pi]$:

$$\begin{aligned} u_t + 4u_{xxxx} + \alpha \left[u_{xx} + \frac{1}{2}(u_x)^2 \right] = 0, \\ u(x, 0) = u_0(x), \quad u(x + 2\pi, t) = u(x, t), \end{aligned} \tag{7}$$

where $x \in [0, 2\pi]$, $t \in \mathbb{R}^+$, and α is the bifurcation parameter.

We refer to the study of the attractors by Hyman and Nicolaenko [46] and follow the approach of Papageorgiou and Smyrlis [47, 48] by considering the initial condition $u_0(x) = -\sin(x)$ to ensure that the integral of the solution over the spatial domain vanishes. Varying the bifurcation parameter α in Equation (7) yields a wide range of attracting solutions such as periodic, bimodal, travelling wave, or chaotic, numerically studied in [46].

We spatially discretise Equation (7) using the Fourier spectral method with the 2/3 dealiasing rule [49] and temporally using the ETDRK4 scheme of Cox and Matthews [50]. We use the stiff partial differential equation integrator [51] in the Chebfun software [52] with a spectral resolution of 512 and a time step of 2.5×10^{-4} to numerically solve Equation (7) for $t \in [0, 10]$. The regimes we considered are listed below based on the values of the bifurcation parameter α:

1. One hundred values of α are uniformly distributed in each of the following intervals: $[18, 22]$, $[23, 33]$, $[43, 45]$, $[56, 65]$, $[95, 115]$. These intervals are chosen to cover a wide range of behaviors according to [46].
2. Five hundred values of α are uniformly distributed in $[120, 130]$.

This leads to a data set of one thousand realisations, equally divided between chaotic and non-chaotic behavior.

Figure 9 shows oscillatory solutions to the KS equation for $\alpha = 20$ (a), 44 (b) and a quadrimodal solution (c). A chaotic solution to the KS equation is depicted in Figure 9 (d). This spatiotemporal chaotic behavior is hard to analyse because of the high dimensionality of the system, i.e. the large number of Fourier modes of the solutions. Thus, we analyse the behavior of the solutions by considering the energy time series

$$\mathcal{E}(t) = \int_0^{2\pi} u(x, t)^2 \, dx, \tag{8}$$

normalized by the transformation $\mathcal{E}(t) \mapsto (\mathcal{E}(t) - m)/(M - m)$, where

$$m = \min_{t \in [0,10]} \int_0^{2\pi} u(x, t)^2 \, dx,$$
$$M = \max_{t \in [0,10]} \int_0^{2\pi} u(x, t)^2 \, dx,$$

such that it lies in the interval $[0, 1]$. The normalized energy time series of the solutions to the KS equation for $\alpha = 20$, 44, 100, and 125 is plotted in Figure 9 (e) to (h), respectively. These figures illustrate the relation between $\mathcal{E}(t)$ and the behavior of the solution $u(x, t)$.

Similarly to Section 4.1, we train the convolutional neural network described in Section 2.2 on the x component of the Lorenz system and test it on the time series from the data set of the KS equation described above. The global accuracy that we obtain to classify the time series between chaotic and non-chaotic is 94.4%. The accuracy for the different classes of attracting solutions in the testing set is reported in Table 3.

We observe that the convolutional neural network classifies correctly time series of bimodal, highly oscillatory, trimodal, and quadrimodal solutions, corresponding to $\alpha \in [23, 33]$, $[43, 45]$, $[56, 65]$, $[95, 115]$, as non-chaotic with an accuracy of 96.4%, 99.2%, 95.4%, and 99.6%, respectively. Moreover, the network gets a score of 99.8% on the set of chaotic time series. However, the energy time series of low-frequency periodic solutions to the Kuramoto–Sivashinsky equation for $\alpha \in [18, 22]$ are misclassified by the neural network since only

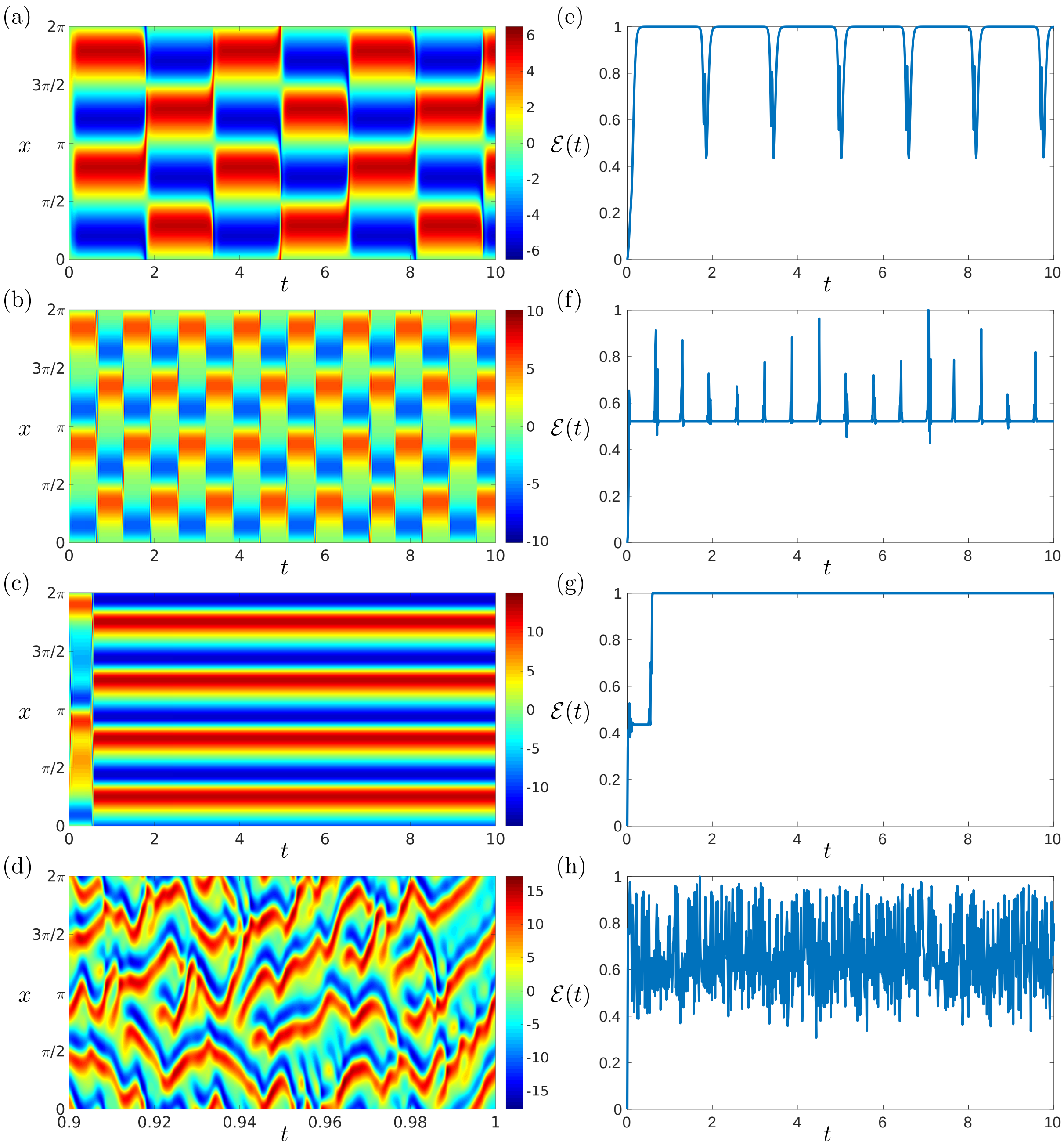

Figure 9: Solutions to the KS equation with $\alpha = 20$ (a), 44 (b), 100 (c), and 125 (d). The right panels show the corresponding normalized energy $\mathcal{E}(t)$. The chaotic solution depicted in (d) has been zoomed to $t \in [0.9, 1]$. (c) and (g) illustrate the transient regime of the solution for $t \in [0, 0.5]$ before convergence to the global quadrimodal attractor.

Table 3: Classification results of the energy time series of the KS equation for various α. The convolutional neural network ConvNet is trained on the x component of the Lorenz system. The classification score is reported for the different intervals of α composing the data set and averaged over five training cycles.

Range of α	Solutions behavior	Accuracy
$[18, 22]$	periodic	54.8
$[23, 33]$	bimodal	96.4
$[43, 45]$	periodic	99.2
$[56, 65]$	trimodal	95.4
$[95, 115]$	quadrimodal	99.6
$[120, 130]$	chaotic	99.8

54.8% of them are identified as non-chaotic. We expect this misclassification to be due to qualitative differences between the corresponding energy time series of the KS equation and the periodic time series of the Lorenz system. In particular, the KS data set contains periodic time series with low frequency oscillations in this regime (see Figure 9 (b)), while the Lorenz system generates periodic time series with high frequency oscillations (see Figure 7 (c)). The neural network is then unable to classify features that are not present in the training set and hence fails to extrapolate to the low frequency periodic time series of the Kuramoto–Sivashinsky equation.

4.3. Accuracy dependence on the training data set size and the time series length

We test the robustness of the convolutional neural network on the classification problem of the KS equation by studying how the accuracy depends on the size of the training data set and length of the time series. Figure 10 shows the effect of the size of the training data set on the classification score of ConvNet. We observe that the neural network achieves an accuracy between 85% and 95% when the the amount of training data available is greater than 10%.

In Figure 11 we analyse the classification ability of ConvNet on shorter time series. The network is trained on time series of the x component of the Lorenz system of same length, whose chaotic classification is obtained using the Lyapunov exponent. The idea here is that the Lyapunov exponent will be estimated less accurately for shorter time series and hence this will affect the accuracy of the network. It is interesting that for all the lengths of the time series we considered the convolutional neural network reaches an average accuracy greater than 80% on the KS problem.

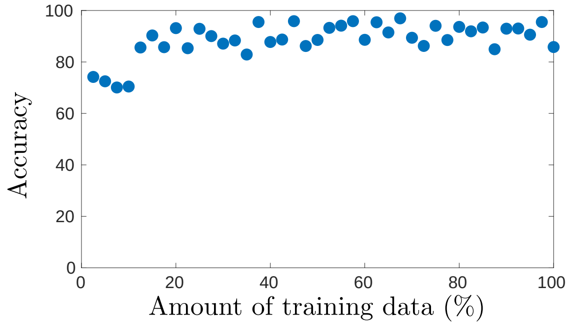

Figure 10: Classification score of the KS data set versus the amount of training data (percentage of five thousand realisations). The classification scores are averaged over ten training cycles of the ConvNet neural network on time series of the x component of the Lorenz system.

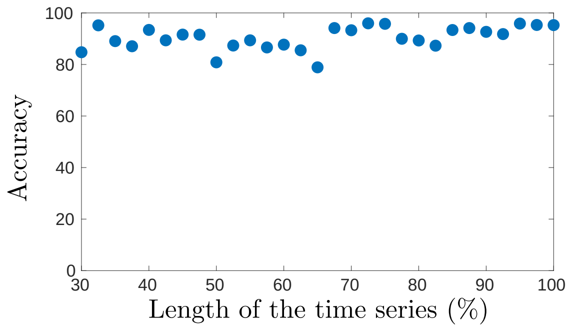

Figure 11: Accuracy of the classification of KS time series with respect to the length of the time series (percentage of one thousand). The classification scores are averaged over ten training cycles of the ConvNet neural network on time series of the x component of the Lorenz system.

Overall, our results show the robustness of ConvNet on this problem. In particular, this network is able to generalize well on time series generated by the KS equation and achieve a classification score greater than 80%, independently of the size of the training data set or length of the time series.

Conclusions

For some of the most challenging real-life applications the expression of a precise underlying dynamical system is unknown or the phase space of the system is infinite dimensional, which makes the identification of the different dynamical regimes in phase space unfeasible or in the best case scenario computationally expensive. For this reason, in this study we have introduced a deep learning approach for classifying time series generated by discrete and continuous dynamical systems. Our approach is to train our neural network on a given dynamical system with a basic or low-dimensional phase space and generalise by using this network to classify time series of a dynamical system with more intricate or high-dimensional phase space.

The proposed convolutional neural network with large kernel size is able to learn the chaotic features of these systems and classify with high accuracy while state-of-the-art neural networks tend to overfit the training data set. In detail, our approach has been applied to classify time series generated by the sine-circle map and the Kuramoto–Sivashinsky equation, using the logistic map and the Lorenz system as training data sets, respectively. We observed a classification accuracy greater than 80% on both systems despite the inability of the network to classify time series with features that are not present in the original training set. Finally, this study suggests that deep learning techniques, which can generalise the knowledge acquired from a training set to a different testing set, can be valuable to classify time series obtained from real-life applications.

There are many directions in which the present results can be pursued further. First of all, attempting to classify time series obtained from real-life applications is crucial. On that respect, the effect of noise in the training and testing data sets is an important aspect to be considered and study the influence of the noise to the accuracy of the networks to classify the time series. We hope to address such an analysis in our future work.

Acknowledgments

The authors would like to thank Samuel Cohen for useful discussions. This work was supported by the EPSRC Centre For Doctoral Training in Industrially Focused Mathematical Modelling (EP/L015803/1) in collaboration with Culham Centre for Fusion Energy. This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 and 2019-2020 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

References

[1] M. Casdagli, Nonlinear prediction of chaotic time series, Physica D 35 (3) (1989) 335–356.

[2] Y.-C. Lai, N. Ye, Recent developments in chaotic time series analysis, Int. J. Bifurcat. Chaos 13 (6) (2003) 1383–1422.

[3] S. Mukherjee, E. Osuna, F. Girosi, Nonlinear prediction of chaotic time series using support vector machines, in: Proceedings of the 1997 IEEE Signal Processing Society Workshop, IEEE, 1997, pp. 511–520.

[4] G. Sugihara, Nonlinear forecasting for the classification of natural time series, Philos. T. R. Soc. A 348 (1688) (1994) 477–495.

[5] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, NIPS, 2012, pp. 1097–1105.

[6] Y. LeCun, Y. Bengio, The handbook of brain theory and neural networks, MIT Press, 1998.

[7] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE international conference on computer vision, IEEE, 2015, pp. 1026–1034.

[8] J. B. Elsner, A. A. Tsonis, Nonlinear prediction, chaos, and noise, B. Am. Meteorol. Soc. 73 (1) (1992) 49–60.

[9] T. Kuremoto, S. Kimura, K. Kobayashi, M. Obayashi, Time series forecasting using a deep belief network with restricted Boltzmann machines, Neurocomputing 137 (2014) 47–56.

[10] Z. Wang, W. Yan, T. Oates, Time series classification from scratch with deep neural networks: A strong baseline, in: International joint conference on neural networks, IEEE, 2017, pp. 1578–1585.

[11] J. Han, A. Jentzen, E. Weinan, Solving high-dimensional partial differential equations using deep learning, Proc. Nat. Acad. Sci. 115 (34) (2018) 8505–8510.

[12] J. Sirignano, K. Spiliopoulos, DGM: A deep learning algorithm for solving partial differential equations, J. Comput. Phys. 375 (2018) 1339–1364.

[13] E. Weinan, B. Yu, The Deep Ritz method: A deep learning-based numerical algorithm for solving variational problems, Commun. Math. Stat. 6 (1) (2018) 1–12.

[14] M. Raissi, Deep hidden physics models: Deep learning of nonlinear partial differential equations, J. Mach. Learn. Res. 19 (1) (2018) 932–955.

[15] M. Raissi, G. E. Karniadakis, Hidden physics models: Machine learning of nonlinear partial differential equations, J. Comput. Phys. 357 (2018) 125–141.

[16] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.

[17] S. H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz, Data-driven discovery of partial differential equations, Sci. Adv. 3 (4) (2017) e1602614.

[18] J. Pathak, B. Hunt, M. Girvan, Z. Lu, E. Ott, Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach, Phys. Rev. Lett. 120 (2) (2018) 024102.

[19] J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, E. Ott, Us-

ing machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data, Chaos 27 (12) (2017) 121102.

[20] H. F. Nweke, Y. W. Teh, M. A. Al-Garadi, U. R. Alo, Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges, Expert Syst. Appl. 105 (2018) 233–261.

[21] T. L. Nwe, T. H. Dat, B. Ma, Convolutional neural network with multi-task learning scheme for acoustic scene classification, in: Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, IEEE, 2017, pp. 1347–1350.

[22] G. A. Susto, A. Cenedese, M. Terzi, Time-series classification methods: review and applications to power systems data, in: Big data application in power systems, Elsevier, 2018, pp. 179–220.

[23] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, P.-A. Muller, Deep learning for time series classification: a review, Data Min. Knowl. Disc. (2019) 1–47.

[24] M. F. Møller, A scaled conjugate gradient algorithm for fast supervised learning, Neural networks 6 (4) (1993) 525–533.

[25] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551.

[26] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, P. IEEE 86 (11) (1998) 2278–2324.

[27] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016.

[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.

[29] Z. Wang, Github repository, `https://github.com/cauchyturing/UCR_Time_Series_Classification_Deep_Learning_Baseline` (2019).

[30] M. Lin, Q. Chen, S. Yan, Network in network, arXiv preprint arXiv:1312.4400.

[31] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: Proceedings of the 32nd International Conference on Machine Learning, PMLR, 2015, pp. 448–456.

[32] R. M. May, Simple mathematical models with very complicated dynamics, Nature 261 (1976) 459–467.

[33] S. H. Strogatz, Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering, Westview Press, 2014.

[34] R. C. Hilborn, Chaos and nonlinear dynamics: an introduction for scientists and engineers, Oxford University Press on Demand, 2000.

[35] C. Sparrow, The Lorenz equations: bifurcations, chaos, and strange attractors, Vol. 41, Springer Science & Business Media, 2012.

[36] E. N. Lorenz, Deterministic nonperiodic flow, J. Atmos. Sci. 20 (2) (1963) 130–141.

[37] Y. Kuramoto, Diffusion-induced chaos in reaction systems, Prog. Theor. Phys. Supp. 64 (1978) 346–367.

[38] Y. Kuramoto, T. Tsuzuki, On the formation of dissipative structures in reaction-diffusion systems: Reductive perturbation approach, Prog. Theor. Phys. 54 (3) (1975) 687–699.

[39] Y. Kuramoto, T. Tsuzuki, Persistent propagation of concentration waves in dissipative media far from thermal equilibrium, Prog. Theor. Phys. 55 (2) (1976) 356–369.

[40] G. I. Sivashinsky, Nonlinear analysis of hydrodynamic instability in laminar flames–I. Derivation of basis equations, Acta Astronaut. 4 (11-12) (1977) 1177–1206.

[41] G. I. Sivashinsky, On flame propagation under conditions of stoichiometry, SIAM J. Appl. Math. 39 (1) (1980) 67–82.

[42] G. I. Sivashinsky, Instabilities, pattern formation, and turbulence in flames, Annu. Rev. Fluid Mech. 15 (1) (1983) 179–199.

[43] D. J. Benney, Long waves on liquid films, J. Math. Phys. 45 (1-4) (1966) 150–155.

[44] A. P. Hooper, R. Grimshaw, Nonlinear instability at the interface between two viscous fluids, Phys. Fluids 28 (1) (1985) 37–45.

[45] G. I. Sivashinsky, D. M. Michelson, On irregular wavy flow of a liquid down a vertical plane, Prog. Theor. Phys. 63 (1980) 2112–2114.

[46] J. M. Hyman, B. Nicolaenko, The Kuramoto–Sivashinsky equation: a bridge between PDE's and dynamical systems, Physica D 18 (1-3) (1986) 113–126.

[47] D. T. Papageorgiou, Y. S. Smyrlis, The route to chaos for the Kuramoto–Sivashinsky equation, Theor. Comp. Fluid Dyn. 3 (1) (1991) 15–42.

[48] Y. S. Smyrlis, D. T. Papageorgiou, Predicting chaos for infinite dimensional dynamical systems: the Kuramoto–Sivashinsky equation, a case study, Proc. Nat. Acad. Sci. 88 (24) (1991) 11129–11132.

[49] S. A. Orszag, On the elimination of aliasing in finite-difference schemes by filtering high-wavenumber components, J. Atmos. Sci. 28 (6) (1971) 1074–1074.

[50] S. M. Cox, P. C. Matthews, Exponential time differencing for stiff systems, J. Comput. Phys. 176 (2) (2002) 430–455.

[51] H. Montanelli, N. Bootland, Solving periodic semilinear stiff PDEs in 1D, 2D and 3D with exponential integrators, arXiv preprint arXiv:1604.08900.

[52] T. A. Driscoll, N. Hale, L. N. Trefethen, Chebfun guide (2014).