Master's Thesis

AI-based Wave Prediction in Complex Plasmas
KI-basierte Wellenvorhersage in komplexen Plasmen

Department of Physics
Ludwig-Maximilians-Universität München

Yanzheng Shen

Munich, April. 21th, 2022

Submitted in partial fulfillment of the requirements for the degree of M. Sc.
Supervised by Dr. Christoph Räth

Contents

1 Introduction

Various natural systems, ranging from physics (weather [33], ocean circulation [2], acoustic chaos, etc.) to biology and medicine (population dynamics [6], heart beat [27], etc.), involve complex nonlinear dynamics. Although these systems are often hard to predict, several methods have been developed and could achieve forecasting, at least at short time scale. Especially after the discovery of neural networks, these techniques are widely used as a tool for analyzing and forecasting nonlinear systems. For example, the artificial neural network (ANN) has been shown to be valuable tools for modeling and forecasting nonlinear time series [36]. Later the recurrent neural network (RNN) was developed. RNNs are networks that contains loops in the structure, which makes it possible to capture the flow of information between different time steps. Consequently, RNNs perform better than ANNs in predicting time series, but, due to the vanishing-gradient-problem, even with the introduction of Long-short-term-memory (LSTM), the backpropagation process still requires a huge amount of time for training. In addition to these neural networks, there has been much attention towards the framework of Reservoir Computing (RC). RC was first discovered as Liquid State Machines (LSM) [18], as Echo State Networks (ESN) by Jarger [10] and as Fractal Prediction Machine (FPM) by Tino and Dorffner [31]. During the training process, only a memoryless, usually linear readout is trained while the weights of the recurrent connections are kept constant. Thus, RC only requires a simple one-step linear regression procedure, which makes RC extremely CPU-efficient and easy to train compared to the other RNNs.
At the same time, RC is able to deliver comparable and in some cases even better results. In the past research, it has been successfully applied to forecast of non-linear chaotic systems, such as Lorenz 96 [3], Mackey-Glass system [11], convection turbulence [25] and Kuramoto-Sivashinsky (KS) equation [26]. However, most of the works are based on synthetic, simulated data for training and testing. In this work, we will demonstrate as one of the first application of RC-based forecasting of real data in complex plasmas.
Complex plasmas consist of micro-particles embedded in a low-temperature plasma [19]. The micro-particles are trapped in experimental plasma chambers. In experiments, high-speed video cameras record the positions of the micro-particles and the motion are calculated from frame to frame. The motions are used as a model system on the kinetic level [21] for various collective effects such as dust density waves (DDWs), which is the data we used for analyzing and forecasting. DDWs form spontaneously in complex plasmas in the presence of an ion flux relative to the micro-particle cloud due to the two-stream instability if the neutral gas pressure is low enough [20]. They can be observed as alternating regions of higher micro-particle density (wave crests) and lower density (troughs) travelling in the direction of the ion flux. At very low gas pressures, the waves become strongly nonlinear [14].
The focus of this thesis is to develop a model that could capture the long-term behavior (its "climate") and make a precise short-term prediction. The very first step is to determine which network to use. There is much research that demonstrate the performance of short-term predictions of different ML networks [3], but the performance on long-term behavior is the subject of comparatively little research. The long-term performance is evaluated by means of the largest Lyapunov exponent and the correlation dimension. To

compensate for the high variance in results, we create large statistics in our numerical experiments.

We mainly deal with two aspects: Firstly, we examine the performance on spatial movement of plasma wave. Since the wave mostly move downward, the data on the top is provided to the network at each frame. The network needs to forecast the movement based on these values. We found that the network can capture the motion with high precision. Since the motion of the wave seems to vary at different regions, we have also applied cross validation (i.e., the model is trained with data from one region, but tested with another region). Essentially, it could show whether the underlying mechanism between different region is the same.

Secondly, we analyzed the performance of the short- and long-term forecast. There are two major steps that will influence the performance. To begin with, we applied the delay reconstruction and moving average to the data. By applying moving average, it changes the amplitude distribution of the wave, which makes it easier for the model to capture the major behavior. Then, we investigated the hyperparameters of the model and numerically tested the performance on reproducing the climate of plasma wave and found that with proper choice of hyperparameters the model successfully captures the statistics. Among all the nonlinear, chaotic behaviors of dynamical systems, the extreme events play an important role in the systems. As these events occur rarely, they are even harder to predicted. However, since these events usually have major effect to the system, the forecast of such events is very important and valuable. Consequently, we also applied the model to short-term forecast of extreme events. We chose the rogue wave as the target (i.e., wave with significantly large height).

Rogue waves, also known as freak waves, were first introduced in Oceanography [23]. They have exponentially large crest-to-trough height exceeding two times the significant height. The significant height (Hs) is defined as the mean of the largest third of the wave heights. Rogue waves are distinct from the Tsunamis, which is caused by the other phenomena, such as earthquake, and do not have a single distinct cause, which makes it hard to predict. Although these waves appear rarely, they could have dramatic influence on ships.

Rogue waves may also appear in other media other than water. For example, in optics [30] and Bose–Einstein condensation [38]. These events in complex plasmas [22] were first reported experimentally by Tsai et al. [32]. They are defined with similar concept as in the Oceanography. The amplitude of the wave height in Complex Plasmas is measured as the maximal micro-particle number density in the wave crest, which in turn can be approximated with the increase in average pixel brightness [29]. Lin and I [13] demonstrated that the formation of rogue waves in Complex Plasmas is associated with the synchronization of 3D particle focusing by preceding distorted waveforms at different scales.

The model is also tested with task specifically designed for rogue waves. The testing set are generated a few frames before the event and the number of successful captures of the events are recorded. We found that the model can capture the rogue wave within a short period of time.

This thesis is organized as follows: In chapter 2, the methods used for later experiments are introduced and we made a comparison between three different machine learning techniques. In section 2.1 - 2.2, the networks (ANN, LSTM) are briefly introduced. Section

2.3 introduces the frameworks of RC and basic implementation which is mainly used for later works. In section 2.4, we explain the measures used to evaluate the performance of long-term simulation and prediction. The last section (2.5) gives an evaluation of the three methods introduced in 2.1 -2.3 by comparing the performance of long-term predictions. Also, the result justifies our decision of using RC for later predictions.
In chapter 3, we developed a hybrid ESN used for spatial prediction of plasma wave. In section 3.1 and 3.2, we describe the experimental and reservoir setup behind the results in section 3.3.
In chapter 4, we adjusted the hyperparameter of ESN and used it for forecasting time series of plasma wave. Section 4.1 describes the motivation and theory used for forecasting in later sections. In section 4.2 and 4.3, the results for short-term and long-term prediction of plasma wave are demonstrated.
In chapter 5, the RC developed in chapter 4 is further tuned, so that it could better capture the feature of rogue wave events. Section 5.1 briefly describes the definition of rogue wave. In section 5.2, we explained the measures used to determine whether the predictions capture the long-term statistics of rogue wave events. We tested the performance of short-term prediction of rogue wave events in section 5.3.
Finally, we conclude by discussing the results of our work in chapter 6.

2 Method

2.1 Deep forward neural network

Neural networks (NN) are techniques of machine learning which could be successfully applied to different applications, such as, computer vision [37], speech recognition [24], data filtering. One of the advantages is that they could be completely data driven. NNs could be trained and learn the structure without a complete knowledge about underlying procedure.
ANN was inspired by the biological neural networks that constitute animal brains. The network contains a collection of neurons or nodes that take, process and transmit a signal with other neurons in the network. Each neuron is linked to the others with different weights. Mathematically, the output of a neuron can be computed as $y = f(\Sigma_i W_i x_i + b)$, where W_i is the trainable weight of input x_i, b is the bias and f is the activation function of the neuron. During the training process, weights and bias of each neuron are adjusted using backpropagation algorithm. It computes the gradient of the loss function of each neuron by the chain rule. In practice, it iteratively computes the gradient of one layer backward in time from the last layer (the output layer in most of the cases), so that redundant calculations could be avoided for intermediate terms in chain rule. The governing equations for a network with L layers are shown below.

$$\delta^L = \nabla_a \mathbf{C} \odot f'(\mathbf{y}^L)$$
$$\delta^l = ((\mathbf{w}^{l+1})^T \delta^{l+1}) \odot f'(\mathbf{y}^l)$$

where $\mathbf{y}^l$, $\mathbf{x}^l$, δ^l is the input, output vector and gradient at layer l, $\mathbf{C}$ is the cost function, $\odot$ denotes the hadamard product. The first equation calculates the gradient at the last layer L. The gradient of the previous layer is related to the current one by the second equation.
Another important concepts in training ANNs are the learning rate and regularization. The learning rate defines the step size of each iteration that the model takes to adjust the errors. Higher learning rate could shorten the time needed for training but could also leads to lower accuracy. In extreme cases, the learning process could also completely fail. On the other hand, a lower learning rate takes longer time for training, but gives the potential for greater accuracy. The regularization prevents the model from the problem of overfitting. Overfitting refers to the phenomenon where the network models the training data very well but fails when it is tested with new data from the same problem domain. The major cause of this problem is the noise in the training set. When the network is trained too well, the noise within the training set can infect the model with substantial errors and reduce its predictive power. Normally, the problem of overfitting often refers to a very complex network with extremely large weights of neurons. The regularization simply uses these phenomena and introduces an extra term in the loss function to prevent the aggregate weight of neurons becomes too larger. Eventually, it leads to a less complex network and solves the problem of overfitting. A mathematical description of the regularization term will be defined in section 2.3.
In forward neural networks, the signal only flows in one direction (i.e., from input to hidden and hidden to output). No loops exist in this kind of network. The advantage of

forward network is that it is easy to design and determine the function of each neuron and layer. However, no hidden variable exists to keep tracking the temporal evolution, which makes it stateless, unlike the other two networks that will be introduced in the next sections. Consequently, the forward neural network often does not perform well in time series prediction.

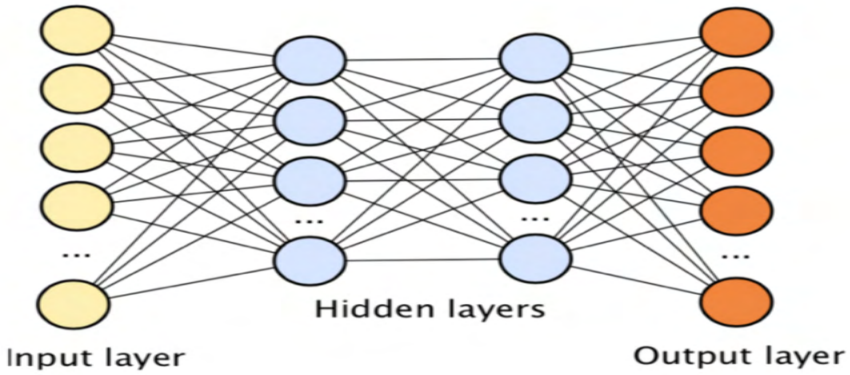


Figure 2.1: A schematic diagram of ANN.

2.2 Recurrent neural network with Long short-term memory

The LSTM was first introduced in 1997 [8], which is a kind of recurrent neural network (RNN). Unlike ANNs, RNNs contain cycles that allow the variable at future steps to capture influences of the present value. This property makes RNNs suitable for predicting sequential data and time series. The computation is governed by equations with expression:

$$h(t) = f_h(\mathbf{W}_{hi}x(t) + \mathbf{W}_{hh}h(t - \Delta t) + b_h) \tag{2.1}$$

$$y(t) = f_{out}(\mathbf{W}_{oh}x(t) + b_y) \tag{2.2}$$

where $h(t) \in \Re^{d_h}$, $x(t) \in \Re^{d_x}$, $y(t) \in \Re^{d_y}$ are the value for hidden unit, input and output at time t. The weights for connections between the input, hidden and output layers is captured by $\mathbf{W}_{hi} \in \Re^{d_h \times d_x}$ (input-to-hidden), $\mathbf{W}_{hh} \in \Re^{d_h \times d_h}$ (hidden-to-hidden), $\mathbf{W}_{out} \in \Re^{d_h \times d_x}$ (hidden-to-output), $b_y \in \Re^{d_y}$ and $b_h \in \Re^{d_h}$ are the biases for the output and hidden layers.

There are also some problems with RNNs. First, it is a long computation and is not parallelizable, which makes it very expensive for training. Besides, the long-term dependencies was the major challenge. Since the training process also use the back-propagation and the computation is always done in finite-precision numbers, the long-term gradient is likely to trend to 0 (vanish) or trends to infinity (explode). However, LSTM successfully copes with the vanishing gradient problem to an extend by using "*gates*".

In LSTM, "*gates*" are trainable variables used to decide whether old values should be forget. The governing equations for LSTM are:

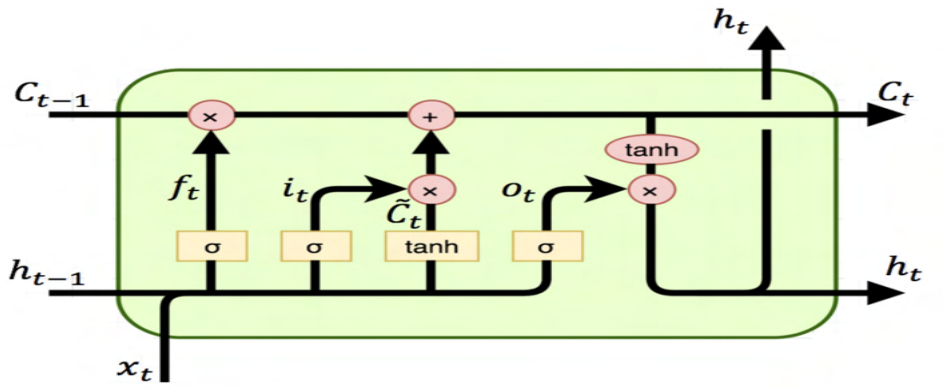


Figure 2.2: A schematic diagram of LSTM. This procedure is repeated at different time frames. f_t, i_t, o_t represent the *gates* for forget, input and output at each time frame.

$$
\begin{aligned}
g^f(t) &= \sigma_f(\mathbf{W}_f[h(t-1), x(t)] + b_f) \\
g^i(t) &= \sigma_i(\mathbf{W}_i[h(t-1), x(t)] + b_i) \\
\tilde{C}(t) &= tanh(\mathbf{W}_c[h(t-1), x(t)] + b_h) \\
C(t) &= g^f(t)C(t-1) + g^i(t)\tilde{C}(t) \\
g^o(t) &= \sigma_h(\mathbf{W}_h[h(t-1), x(t)] + b_h) \\
h(t) &= g^o(t)tanh(C(t))
\end{aligned}
$$

where $g^f, g^i, g^o \in \Re^{d_h \times (d_h + d_x)}$ define the trainable gates for "forget", "input" and "output", $\sigma_f, \sigma_i, \sigma_h$ are the corresponding activation functions, $C(t)$ is the cell state at time t.
It has been shown that LSTM is successfully applied to prediction of sequential data, such as, speech recognition [15], hand-writing recognition [5] and language translation [9].

2.3 Reservoir Computing and Echo state network

Reservoir Computing, also known as the Echo state network, is a recurrent neuron network which contains a reservoir with D sparsely connected neurons. The connectivity of the neurons is governed by an adjacency matrix $\mathbf{A} \in \Re^{D \times D}$. In this thesis, reservoirs are always generated as an Erdos-Renyi network. The weights of matrix $\mathbf{A}$ are initialized from uniform distribution on interval $[-1, 1]$. Afterward matrix $\mathbf{A}$ is normalized to its maximum eigenvalue and rescaled to some fixed value of spectral radius ρ, which is an essential hyperparameter used in reservoir computing. The spectral radius ρ is often free and need to be optimized according to the experiments and data.
In general, ESNs takes the input $x \in \Re^{d_x}$ and converts it into a higher dimensional reservoir state $r(t) \in \Re^D$ by the input layer $\mathbf{W}_{in}$, which is itself a non-linear dynamical system. The dynamics is governed by the update of $r(t)$ by equation:

$$r(t + \Delta t) = f(\mathbf{A}r(t), \mathbf{W}_{in}x(t)) \tag{2.3}$$

where f is the activation function, $\mathbf{W}_{in} \in \Re^{d_x \times D}$ is the input matrix, $x(t)$ is the input at time t.
$\mathbf{W}_{in}$ is also chosen to be spares, which means there is only one non-zero element in each row. Therefore, each node is only connected to one degree of freedom of the input. The same was done in [17]. There are several choices for activation function f. Here, we will stick to the hyperbolic tangent, which is a sigmoidal function often used in reservoir computing. Initially, f, $\mathbf{W}_{in}$ and $\mathbf{A}$ are determined and kept constant during the training and testing of the network.
After $r(t)$ is calculated, it is fed into the readout layer $\mathbf{W}_{out} \in \Re^{d_y \times D}$, which links the reservoir state to the actual output. The weights of $\mathbf{W}_{out}$ are updated during training and $\mathbf{W}_{out}$ is the only trainable matrix in ESN. During training, the dataset $X_{train} = [x(0), ..., x(T_{train}]$ is passed into the network and $r_{train} = [r(0), ..., r(T_{train})]$ is computed. In order to avoid the influence of the initial condition of the reservoir, the first T_{sync} steps are discarded. For $0 < t < T_{syncs}$, the reservoir is synchronized with the training data. The training process for $\mathbf{W}_{out}$ is done by linear regression with L_2 Regularization (Ridge Regression). $\mathbf{W}_{out}$ can be determined by minimizing the following equation:

$$\sum_{T_{sync}<t<T_{train}} ||\mathbf{W}_{out}\widetilde{r}(t) - v(t)||^2 + \beta||\mathbf{W}_{out}||^2 \tag{2.4}$$

where $v(t)$ is the target output at time t, β is the constant for L_2 norm to control the magnitude of regularization. Note that here $\widetilde{r}(t)$ is used instead of $r(t)$. In most of the cases, $\widetilde{r}(t)$ equals to $r(t)$, but it can also be some non-linear transformation of $r(t)$.
Unlike ANN and LSTM, which requires multiple steps of iterations for the weights of neurons to converge, ESN only requires one-time computation for training equation.2.6. Therefore, ESN is often more efficient than the other neuron networks.

$$\mathbf{W}_{out} = (\widetilde{r}^T\widetilde{r} + \beta\mathbb{I})^{-1}\widetilde{r}^T v \tag{2.5}$$

where $\widetilde{r}$ is r_{train} in matrix form after discarding the synchronization steps and v is analogous.
The next step is to compute the output $y(t) \in \Re^{d_y}$. The aim is to make a prediction such that $y(t) \cong x(t + \Delta t)$ at every timestep. The prediction is done by putting the output

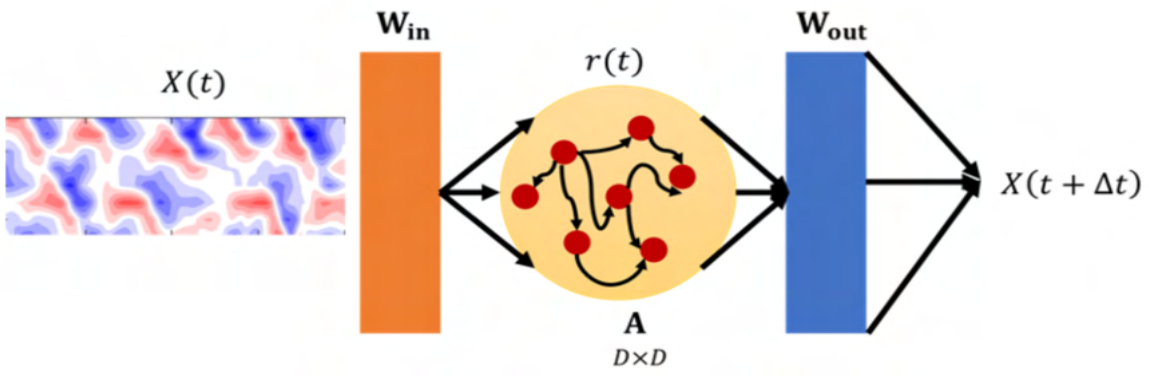


Figure 2.3: A schematic diagram of RC–ESN with $D \times D$ reservoir from [3].

back into the reservoir and forms a closed loop. As a result, the dimension of input and output muss be same, i.e., $d_x = d_y = d$. And the target output $v(t)$ is just $x(t + \Delta t)$. Thus, the equations for the training process can be defined as:

$$r(t + \Delta t) = tanh(\mathbf{A}r(t) + \mathbf{W}_{in}x(t)) \tag{2.6}$$

Then, the output $y(t)$ is computed as:

$$y(t) = \mathbf{W}_{out}r(t) \tag{2.7}$$

Generally, the reason how and under which criteria does the ESN work is not fully understood. The choice of hyperparameters (spectral radius, dimension of reservoir, ...) is also an open question. Among them, the spectral radius ρ is the important one and has been studied in lots of research. It is often assumed that $\rho < 1$ is the condition for ESN to have the echo state properties and make a good prediction. Although $\rho < 1$ often gives a reasonable result and works in practice, it has been found that it is not necessary nor sufficient for ESN. In our research in prediction of plasma wave, we have found that with $\rho > 1$, it achieved a better result.
Another important choice of hyperparameter is the activation function and its readout. It has been found that breaking Symmetry of the reservoir is a crucial procedure for RC [7]. The symmetry means the equations or mechanism is symmetric under certain transformation of coordinates. For example, in Lorenz system, the equation is symmetric under transformation $(x, y, z) \to (-x, -y, z)$. Due to the existence of symmetry, mirror attractors will appear in the system and cause the prediction to jump between them.
Assume $x = \{x_0, ..., x_T\}$ and its inversion $-x = \{-x_0, ..., -x_T\}$, Because of anti-symmetry of the activation function $(tanh)$ the linearity of the readout,

$$\begin{aligned} y(t, -x) &= \mathbf{W}_{out}r(t, -x) \\ &= \mathbf{W}_{out}[tanh(\mathbf{A}r(t-1, -x) - \mathbf{W}_{in}x(t-1))] \\ &= -\mathbf{W}_{out}[tanh(\mathbf{A}r(t-1, x) + \mathbf{W}_{in}x(t-1))] \\ &= -\mathbf{W}_{out}r(t, x) \\ &= -y(t, x) \end{aligned} \tag{2.8}$$

And,

$$\begin{aligned} \mathbf{W}_{out}(-x) &= (r^T(-x)r(-x) + \beta\mathbb{I})^{-1}r^T(-x)(-x) \\ &= (r^T(x)r(x) + \beta\mathbb{I})^{-1}r^T(x)(x) \\ &= \mathbf{W}_{out}(x) \end{aligned} \tag{2.9}$$

Consequently, the training is completely equivalent on x and $-x$.
There are several ways for symmetry breaking, such as, adding a bias term in the readout and the Lu readout. Lu readout is a quadratic extension of the readout introduction by Lu et al. [16], which has the form $\widetilde{r} = r_1, ..., r_D, r_1^2, ..., r_D^2$ in the readout. As a result,

$$\begin{aligned} y(t, -r(t)) &= -\mathbf{W}^1_{out}r(t) + \mathbf{W}^2_{out}r^2(t) \\ &= -y(t, r(t)) + 2\mathbf{W}^2_{out}r^2(t) \end{aligned} \tag{2.10}$$

where $\mathbf{W}^1$ and $\mathbf{W}^2$ are first and second half of $\mathbf{W} \in \Re^{d_y \times 2D}$
In [7], Lu Readout performed significantly better than the others in terms of the forecast horizon of its prediction. Thus, for the experiments in this thesis, we will use Lu Readout as the setup for the ESN.

2.4 Measure

2.4.1 Correlation Dimension

An evaluation of the prediction capabilities must include the statistical long-term properties. One of the important characteristic is the structural complexity, which could be quantified by calculation of correlation dimension. It measures the dimensionality of the space populated by the trajectory of a system, which can be calculated by Grassberger Procaccia algorithm [4].
The basic idea of correlation dimension is that: Let us first define a collection of points $\mathbf{x}_i, i \in 1, ..., n$ on the attractor and the fraction $n_i(\epsilon)$ of all possible points which are closer than a given distance ϵ from a specific point $\mathbf{x}_i$. The approximation can be written as:

$$n_i(\epsilon) = \frac{1}{N-1} \sum_{j=1, j \neq i}^{N} \Theta(\epsilon - |\mathbf{x}_i - \mathbf{x}_j|) \tag{2.11}$$

where Θ is the Heaviside step function. (i.e. $\Theta(x) = 1$ if $x > 0$ and $\Theta(x) = 1$ if $x \leq 0$).
Locally we expect it to scale as a power law, $n_i(\epsilon) \propto \epsilon^{D_i}$. Then the pointwise dimension D_i is defined as

$$D_i = \lim_{\epsilon \to 0} \frac{log\ n_i(\epsilon)}{log\ \epsilon} \tag{2.12}$$

Since the local measurement is different in each part of the attractor, we need to average over the complete set to get the general measure. Therefore, the correlation integral is:

$$C(\epsilon) = \lim_{N \to \infty} \frac{2}{N(N-1)} \sum_{i,j=i+1}^{N} \Theta(\epsilon - |\mathbf{x}_i - \mathbf{x}_j|) \tag{2.13}$$

In the limit of infinite amount of data and small ϵ, the correlation integral is expected to scale like a power law, $C(\epsilon) \propto \epsilon^{\mathcal{V}}$. The correlation dimension V can be defined as:

$$\mathcal{V} = \lim_{\epsilon \to 0} \frac{log\ C(\epsilon)}{log\ \epsilon} \tag{2.14}$$

It is obvious that the limits we need to take in equation 2.13 and 2.14 will leads to troubles when we only have a finite sample. N is limited by the size of sample and ϵ is limited from below by the accuracy of data and the lack of neighbors at small length scale.
To numerically estimate the correlation dimension, one needs to calculate $C(\epsilon)$ in a wide range of ϵ. From the theory, the correlation dimension should be the slope of the $log\ C(\epsilon)$ against $log\ \epsilon$. However, in practice the scaling only holds in particular region of $\epsilon \in [\epsilon_{min}, \epsilon_{max}]$, where ϵ_{min} is bounded below by the minimum distance of two point and ϵ_{max} is bounded by the size of the attractor. Then, the correlation dimension is just the slope,

which could be determine by curve fit with linear regression. Since we do not need high precision, to speed up the process, we use the following equation to approximate the slope.

$$\mathcal{V} = \frac{log\ C(\epsilon_{max}) - log\ C(\epsilon_{min})}{log\ \epsilon_{max} - log\ \epsilon_{min}} \tag{2.15}$$

In general the values for ϵ need to be optimized for different data. For the measurements in this thesis, we used an algorithm to preform this task. We simply did a grid search of ϵ_{min} and ϵ_{max} for the given dataset. More information about the correlation dimension could be found in chapter 6 in [12].

2.4.2 Lyapunov Exponents

Another important measure of the long-term behavior is the temporal complexity of a system. The divergence of a trajectory is not very dramatic if the process is very slow. Thus, we are more interested in the system if the divergence is exponential fast. The exponential divergence could be quantified by Lyapunov Exponents. Besides, there are already some well-known algorithms to calculate them form both the data trajectories and the governing equation for the system. Consequently, even when the origin of the data is unknown, the measure of Lyapunov Exponents is still accessible in practice. In general, there are many different Lyapunov Exponents depending on the phase space dimension of the system. In principle, a system is defined as chaotic if at least one of the Lyapunov exponent is positive. Here we will focus on the most important one, the Maximum Lyapunov Exponent (MLE) λ.
Let us define two points x_0 and x_1 in the space with distance $||x_0 - x_1|| = \delta_0 << 1$. We could consider δ_0 as a perturbation on x_0 and $x_1 = x_0 + \delta_0$. Denote $\phi(x, t)$ as the flow. Then by taking the Taylor expansion of the flow:

$$\phi(x_0 + \delta_0, t) \simeq \phi(x_0, t) + D_{x_0}\phi(x_0, t)\delta_0 \tag{2.16}$$

Since in chaotic system we expect an exponential divergence, the perturbation is defined as:

$$||\delta_t|| = ||D_{x_0}\phi(x_0, t)\delta_0|| = ||\delta_0||exp(\lambda t) \tag{2.17}$$

Therefore, the local value of MLE $\lambda(x_0)$ is defined as:

$$\lambda(x_0) = \lim_{t\to\infty}\lim_{\delta_0\to 0}\frac{1}{t}ln\frac{||\delta_t||}{||\delta_0||} \tag{2.18}$$

If λ is positive, it means that there exists an exponential divergence of trajectories around x_0. In some dissipative systems λ could also be negative which means that two trajectories approach each other exponentially fast. Consequently, it reflects that there exists a fixed stable point. Another possible motion is when $\lambda \simeq 0$, it means that the motion of two trajectories converge or diverge slower than exponential. It is called "Marginal stable", and the motion will form a limited cycle. Table below gives a summary of the motions and corresponding MLE.

motion	Maximum Lyapunov exponent
stable fixed point	$\lambda < 0$
stable fixed cycle	$\lambda = 0$
chaos	$\lambda > 0$
noise	$\lambda = \infty$

The MLE reveals the important characteristic of any chaotic system. Another important feature is that it could be computed easily with Rosenstein-Kantz Algorithm. Here, we will review the basics of the algorithm.

Rosenstein – Kantz Algorithm

The Rosenstein-Kantz algorithm [28] was introduced in 1993. It calculates MLE from the prerecorded trajectories (i.e., measured data). First, it chooses a data-point in the time series as reference point and selects all the points which are within a spatial distance ϵ and have a larger temporal distance than some threshold. The trajectories are likely to be dense in the attractor, if it is long enough. Thus, the different selections of trajectories could be considered as perturbations of each other. Then, compute the distance of the neighbor to the reference point as a function of time. Repeating this process several times with different choice of reference point and time, the noise and local fluctuation could be averaged out.
In practice, it is similar to the procedure of computation of correlation dimension. First, the distances between all pairs of points are calculated. We apply the temporal threshold and select all the neighbor pairs. Then, we compute the mean value of logarithmic distance S of previous neighbor pairs within the range t_{min} and t_{max}. Finally, we use Linear Regression to fit a line to these values. And the MLE is the slope of the line. To speed up the process, we could use the value of $S(t_{min})$ and $S(t_{max})$ to estimate the result, and the MLE is defined as:

$$\lambda = \frac{S(t_{max}) - S(t_{min})}{t_{max} - t_{min}} \tag{2.19}$$

2.5 Comparison between Machine-learning Techniques

In this section, the performance of different machine-learning methods for prediction of long-term behavior (also called "climate") of systems is examined. The methods being examined are echo state network (ESN), recurrent neural network with long-short term memory (LSTM) and deep-forward artificial network (ANN). In paper [3], it demonstrated the performance of predicting short-term time evolution for these methods and showed that the predicting power of ESN surpasses the other two techniques. We will use similar structures and setups for the networks to examine the prediction of climate. Here, we will first examine the short-term predictions, by measuring the forecast horizon, then the long-term predictions, by the correlation dimension and Maximum Lyapunov Exponents. The well-known chaotic system, "Lorenz System", has been used through this section. It

is defined by the equations

$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= x(\rho - z) - y \\ \dot{z} &= xy - \beta z \end{aligned} \tag{2.20}$$

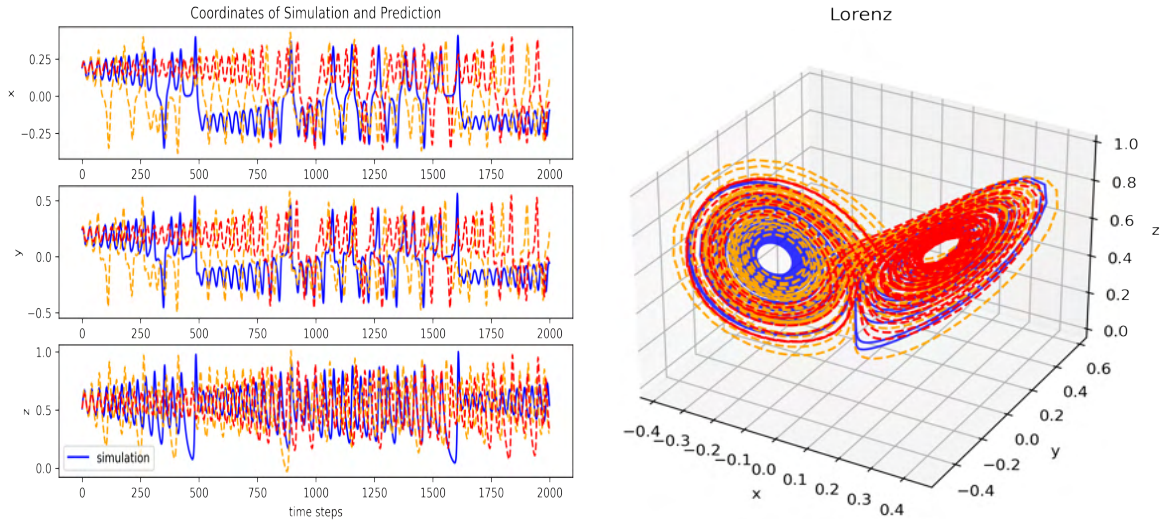
where standard parameters are set to $\sigma = 10$, $\beta = 8/3$ and $\rho = 28$. The trajectory is generated by using the fourth order Runge-Kutta method with timesteps of $\Delta t = 0.02$. In order to examine the prediction of the climate, we calculated the correlation dimensions and Maximum Lyapunov Exponents of the prediction of 10000 timesteps and compared them to the value of Lorenz system. To make the results comparable between different techniques, the activation function for the neurons is set to "tanh" for all three methods. Since the range of "tanh" lies in $[-1, 1]$, the training and testing sets are normalized to have a magnitude within this region.

2.5.1 Feed-forward artificial neural network (ANN)

The training and testing sets are generated as the difference between the current timestep and the next one, i.e. $[(X(t)), (X(t + \Delta t) - X(t))]$.
The network contains 6 fully connected layers, with one input, one output layer with 3 units and 4 hidden layers with 100 hidden units each.
Fig 2.4 shows 2 realizations for the short-term forecast for 2000 timesteps. Both realizations used the same training and testing sets. Only the initial weights of the neurons are chosen randomly. Even though the networks are trained with same data, the predictions still deviate from each other within a short time compared to the other networks, which will be demonstrated in the following sections. The predictions start to deviate from the true values after around 80 timesteps, but the 3D representation of the prediction still looks similar to the Lorenz system. However, we have also tested the short-term behavior with different start of training and testing set and found that the predictions are heavily dependent on the initial point of the training and testing sets. If the start point differs a lot, the prediction of ANN is likely to fall into a different attractor or even fails to reproduce the general structure of Lorenz system, which means that the predictions do not capture the long-term statistics, such as Correlation dimension or Maximum Lyapunov Exponents. This behavior will be shown later with more details in the comparison between different networks.

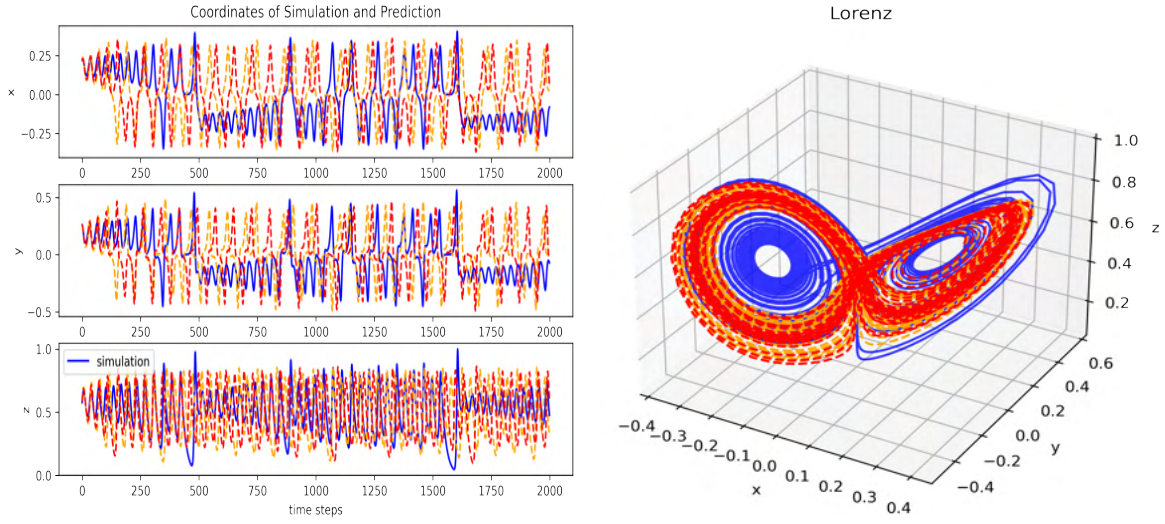
2.5.2 Recurrent neural network with long short-term memory (LSTM)

Since LSTM predicts the next state with several previous states, the training and testing sets are generated as $[X(t - (q - 1)\Delta t), ..., X(t - \Delta t), X(t)] \rightarrow [X(t + \Delta t)]$. For training sets and the first q steps of the prediction the input of the network is from the data set. After q steps the prediction is based on the previous predictions.
The architecture used is similar to the one used in [34]. In this experiment, the network contains just one LSTM layer with 50 hidden units and one output layer (Dense layer) with 3 units as the output. The lookback q is set to 5.
In Fig 2.5, it demonstrates two realizations of predictions. The general setup is like the



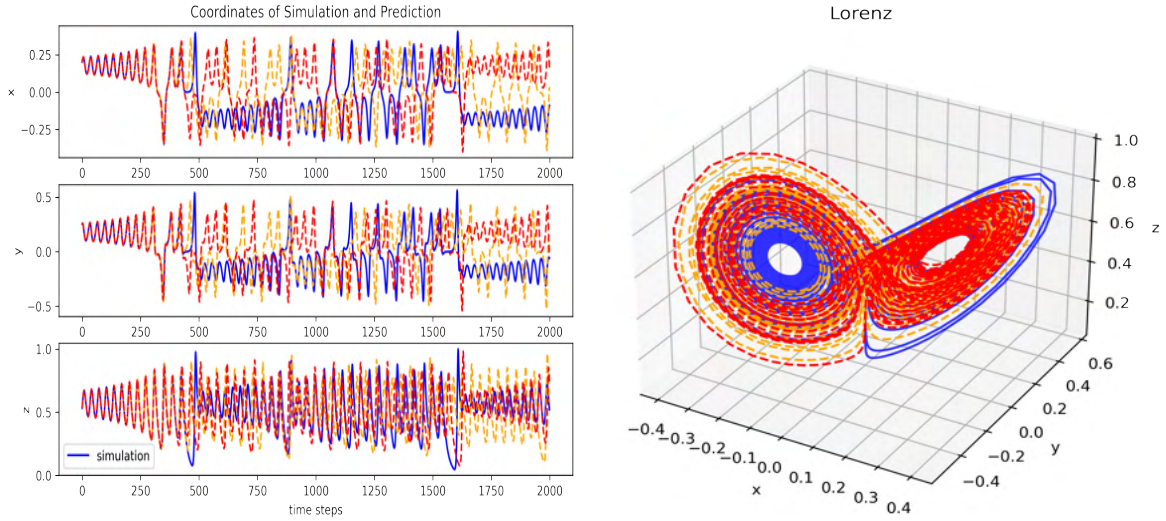
(a) Time evolution (b) 3D representation

Figure 2.4: Short-term forecast for ANN. The lines show real data (blue solid line), two realizations of prediction (dashed lines).



(a) Time evolution (b) 3D representation

Figure 2.5: Short-term forecast for LSTM. The lines show real data (blue solid line), two realizations of prediction (dashed lines).



(a) Time evolution (b) 3D representation

Figure 2.6: Short-term forecast for ESN. The lines show real data (blue solid line), two realizations of prediction (dashed lines).

ANN in last section, where the training and testing sets are the same as ANN. Both predictions have very similar trajectories, even after 2000-time frames. Other predictions also show similar behavior, which means that LSTM give more stable results than ANN.

2.5.3 Echo state network (ESN)

During training, the dimension of the reservoir (A) is set to 1000 with spectral radius ρ of 0.1 and average network degree of 100. W_{in} and A are initialized with random numbers. For prediction, the first 500 timesteps are synchronized. Then, it made a prediction of 10000 timesteps based on the output of the synchronization. We have further noticed that the performance of the climate prediction is for $0.1 < \rho < 0.8$ not sensitive to the spectral radius ρ.
Fig 2.6 demonstrates two realizations of prediction of ESN. The training and testing set are the same one used for ANN and LSTM. As describe in previous section, ESN requires a process called "synchronization" to get rid of the influence of the initial condition. Therefore, the first 200 time frames are discarded. To make the result more comparable between different methods, the "synchronization" started 200 frames before the testing sets, so that the predictions are based on the same data as the ANN and LSTM. The results shows that prediction only start to deviate from the real values after 400 frames.

2.5.4 Comparison

We first examined the short-term forecasts by measuring the forecast horizon. It is defined as the time between the start of a prediction and the point where it deviates from the

Method	Forecast horizon in Δt
ANN	12.3 ± 8.9
LSTM	49.2 ± 15.6
ESN	108.7 ± 21.5

Table 1: Performance of short-term prediction for different Machine Learning methods on Lorenz system.

test data more than a fixed threshold as in [7]. The exact condition is shown below:

$$|\mathbf{v}(t) - \mathbf{v}_R(t)| > \boldsymbol{\delta} \tag{2.21}$$

where the norm is taken elementwise and $\mathbf{v}(t), \mathbf{v}_R(t)$ represent the prediction and testing data, $\boldsymbol{\delta}$ is the threshold, which has same dimension as $\mathbf{v}(t)$. Here, we used $\boldsymbol{\delta} = (5.8, 8.0, 6.9)^T$ for Lorenz system.
Since we have found that the initial condition of the datasets will affect the performance of the prediction. To make a comparison between different networks, we generated the training and testing randomly for each realization, so that the influence of distinct choices of the datasets could be examined. The training and testing sets are generated such that these is no overlap between them.
For each algorithm, we systematically investigated this effect by running the network with 100 different random number seeds. The mean values of forecast horizon obtained using different algorithms are shown in Table 1. Here, we also found that ESN obtains the largest forecast horizon (about 108 Δt). The LSTM obtains the next-best result with around 49 Δt. ANN only obtains around 12 Δt before the predicted trajectories deviate from the testing data. The performances are similar to result shown in [3].
Then, we compared the long-term predictions. In Fig 2.7, it demonstrates the maximum Lyapunov exponent and correlation dimension of 10000 timesteps predictions for ESN, LSTM and ANN. Each network was trained with 100 realizations with randomized initial conditions (training, testing sets and initial weights of neurons), but the hyperparameters are kept the same (number of layers, nodes, size of reservoir, etc.). The red dot shows the correct value calculated from 50 simulations of Lorenz system; the error bar is five times the standard deviation from these simulations.
The results show the predictions for ESN and LSTM almost all lie within the error bar. For LSTM, the means are slightly deviated from the true values, but still within a reasonable range. For ANN, although most of the predictions lie around the correct value, there are about 30% of prediction that had a correlation dimension around 0 and failed in capture the statistics of the system. For instance, some of these failed predictions have strong decay on the amplitude and become a constant function after several timesteps. This is expected, since ANN is stateless and does not keep the temporal motion of the system. Consequently, LSTM and ESN surpass ANN in capturing the long-term behavior of dynamical systems.

In practice, the running time required for each method is also an important feature, since the training process often involve large amount of data and could be expensive to compute. To better understand how well these techniques works in practice, we have also

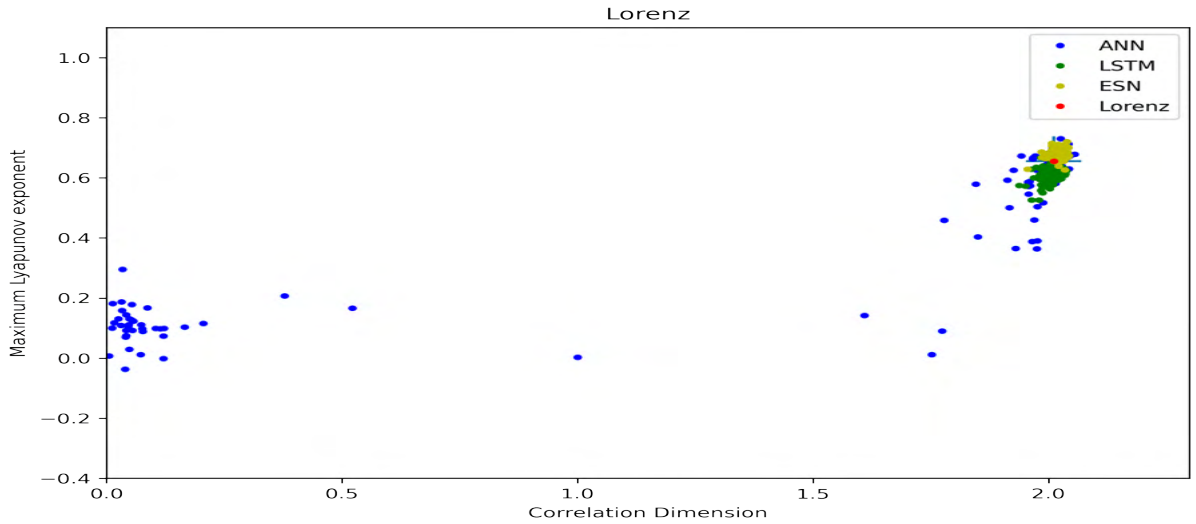


Figure 2.7: Maximum Lyapunov exponent scattered against the correlation dimension for different ML techniques. Red dot shows the correct values for Lorenz system. The error bar corresponds to five times the standard deviation.

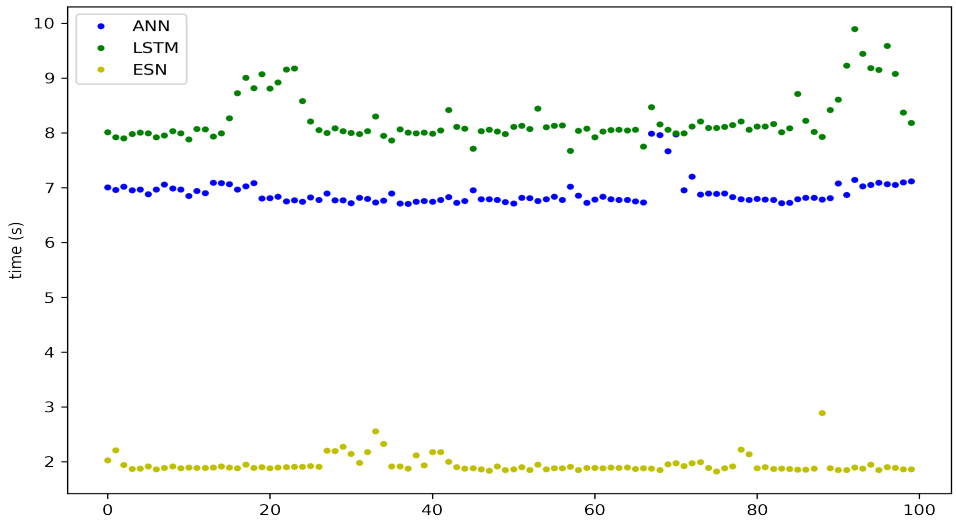


Figure 2.8: Length of time taken for training and testing for each ML method.

measured the CPU-time required to train each network. As we have mentioned earlier, ESN only requires a one-step linear regression on the readout, so it is expected to be the most efficient method. The running time for ANN and LSTM depends on the number of iterations and complexity of the network (number of layers and nodes). With a large number of training iterations, the network could capture the features of the training set with high precision, but it will definitely cost more training time. Similar for the network complexity, a network with more nodes and layers will be able to capture higher order features of the data, but it will be computationally expensive. For ANN and LSTM, we started to train the networks with small number of iterations and evaluated the performance on short-term and long-term prediction on the testing sets. The testing error will first decline, and then after a certain number of iterations, the error almost does not change. Due to the problem of overfitting, it might increase again, if the network is very complex (i.e., with many hidden units and layers). The number of iterations for training is chosen such that an increase of number of iterations does not give significant difference on the testing set. We also expect LSTM to be slower than ANN, as there are more parameters for training.
The results are shown in Fig 2.8. On average, ESN required the least time in training and predicting among all three methods. However, we have also observed some realizations for ESN that fail to make a prediction and cost a huge amount of time. It is caused by the occurrences of error in calculation of the eigenvalues of reservoir. Since the reservoir and input matrices are randomly generated, it could lead to numerical error when calculating the eigenvalues and inverse of the matrices. These numerical errors do not happen frequently. From our experiments, the chance that these errors occur is less than 1%. In most of the cases, the training process of ESN is stable.

2.5.5 Summary

In this section, we evaluated the performance of different networks. For short-term prediction, we found that found that ESN surpasses LSTM and ANN. And the recurrent networks (LSTM and ESN) could reproduce the climate of Lorenz system with stable results. Yet, the forecast with forward-feeding networks (ANN) is not as stable as the recurrent networks. Though there are more results that capture the general structure, about 30% of the realizations from ANN fails to capture the climate. More sophisticated structure of ANN might be able to improve the performance and stabilize the result, but a larger and more complex network will be expensive to training.
The performances of ESN and LSTM are similar and both methods successfully reproduce the long-term behaviors of Lorenz system. However, ESNs are more efficient than LSTM. As shown in Fig 2.8, it only takes about one-fourth of the time needed to train a LSTM network. Therefore, ESN is used for the later experiments in this thesis.
In general, there is also other more advanced deep-learning RNNs that outperform the simple LSTM. Since these advanced techniques could be a topic on its own and out of the scope of this thesis. And, as mentioned before, these techniques often link to a sophisticated network which is computational expensive. Here, we are satisfied with the results for comparison of simple networks as ANN and LSTM.

3 Spatial Prediction of Plasma Wave

3.1 Experimental Setup

The experiment is performed by research group at DLR. Here, we will just give a brief introduction about the setup. More details about the experimental setup could be found in [1].
The Dust density waves (DDW) was generated by the ground-based PK-3 chamber. The experimental setup is shown schematically in Fig 3.1. A plasma was created in a capacitively-coupled RF chamber. The two electrodes were driven by a sinusoidal signal of 13.56 MHz. The experiment used argon at a pressure of 23 Pa and injected monodisperse melamine formaldehyde (MF) spheres of mass density 1510 kg/m^3 and diameter 2.15 μm. A laser with a wavelength of 686 nm and a power of 45 mW illuminated the micro-particles in a vertical plane. Their motion was tracked using the CMOS video camera Photron Fastcam-1024 PCI at a speed of 500 frames per second (fps) with a field of view of 1024 x 1024 pixels^2 at a spatial resolution of 26 μm/pixel. This high-speed imaging makes it possible to study the particle behavior at the kinetic level and track particles individually from frame to frame even inside the high-speed waves studied in this work.
By heating the bottom electrode with eight resistors and cooling the top electrode with

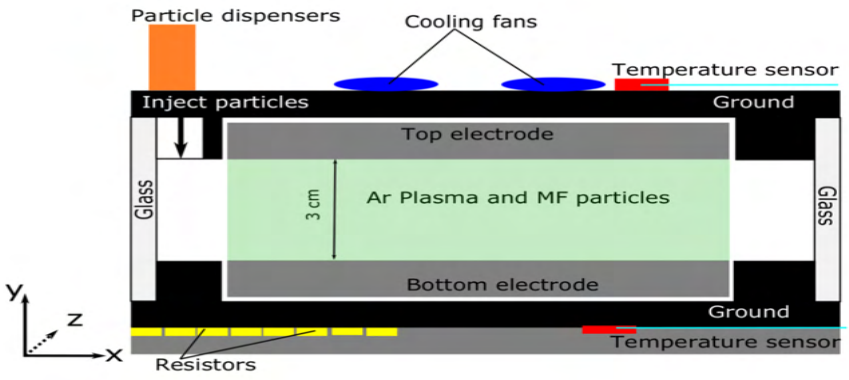


Figure 3.1: A schematic setup of ground-based PK-3 Plus chamber from [1].

two fans, it is able to create a temperature difference between two electrodes. The particles experience a thermophoretic force pointing upwards, against the force of gravity due to this temperature gradient. The magnitude of the force is proportional to the temperature difference. This causes the micro-particle cloud to move away from the bottom electrode into the bulk plasma. One instance of such a raw image is shown in Fig 3.2.

3.2 Reservoir Setup

3.2.1 Data

The dataset is generated by averaging over the intensities of wave at pixels from 10.4 to 18.2 mm of x-axis. Since our model uses hyperbolic tangent as the activation function, the

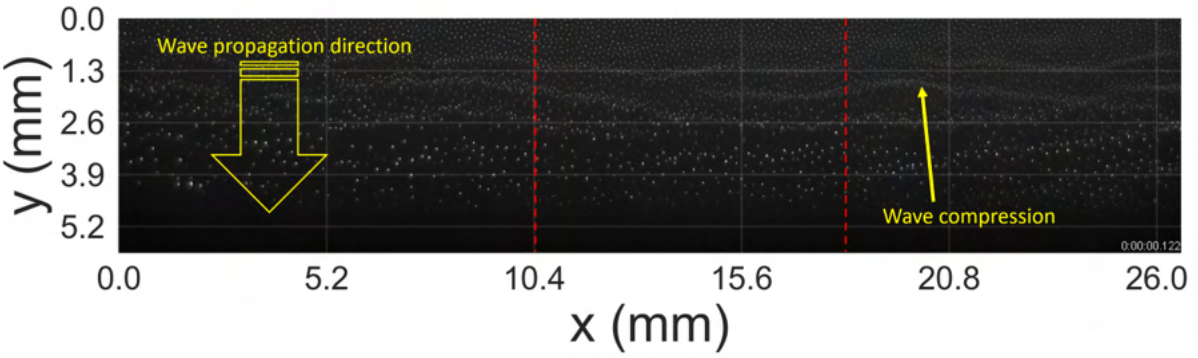
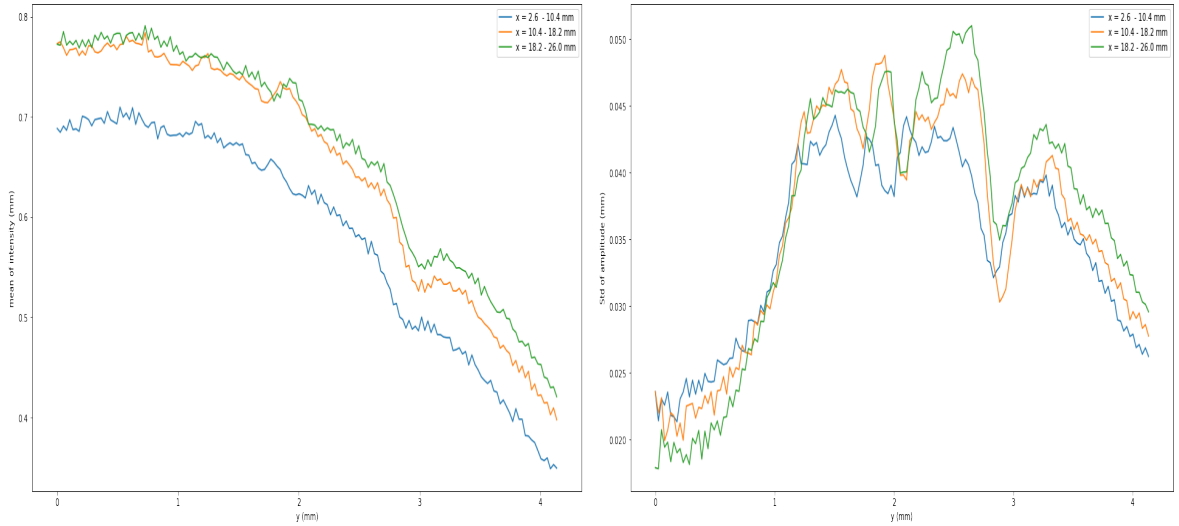


Figure 3.2: Raw image from the high-speed camera at $\Delta T = 4$ K. The wave compression at this particular time instant has also been labeled. The arrow in yellow shows the direction of flow of the DDWs towards the lower electrode, same as the direction of ion-flow. The red dashed vertical lines indicate the x-region in which the waves are further analyzed throughout this work.

training and testing sets are normalized in order to obtain values between -1 (maximum) and 1 (minimum) to enable the usage of the full range of the activation function by the training set.
We first measured the mean intensity and the standard deviation of the amplitude of the plasma wave at each y-pixel, as shown in Fig 3.3. The micro-particles are injected into the system from the top. At small values of y ($y < 1.3$ mm), the plasma wave just starts the formation, so the standard deviation, which represents the average amplitude of the wave, is relatively lower than the other regions. The amplitude starts to increase until around $y = 1.3$ mm, which is about 50 pixels. At $y = 2.6 - 3.12$ mm, there is a sharp decline of the amplitude followed by an increment. Then, the amplitude decreases almost linearly with respect to the y-value. Similar behavior was found for the mean values of intensity, where the rate of decrease with respect to the y-value is smooth in most of the regions. This behavior could also be observed from the period grams (Fig 3.2), where the wave disintegrates around that region. For training and testing, we used the data from the middle region ($x = 10.4$ - 18.2 mm).
To better understand how the wave patterns can be capture, ESNs are trained with different resolutions: first, with the dataset averaged over every 10 pixels (0.26 mm) of y-axis, then, with data at every y-pixel.

3.2.2 Training and Prediction

During training, the dimension of the reservoir is set to 1000 with spectral radius ρ of 0.1 and average network degree of 100. ESNs are trained with 6000 frames (12 s) of a 3-dimensional data of three continues intervals on y-axis. For testing and prediction, at every time step, the first value (smallest y-value) is set to the real data and next two values are predicted except for the initial. Then, the prediction of next time step is based on these values. Eqn 3.1 demonstrates the procedure.



(a) (b)

Figure 3.3: (a) Mean value of the intensity of Plasma wave (b) Standard deviation of intensity at different vertical position (y) for three different x regions.

$$\begin{pmatrix} y_i^t \\ p_{i+1}^t \\ p_{i+2}^t \end{pmatrix} \rightarrow ESN \rightarrow \begin{pmatrix} p_i^{t+1} \\ p_{i+1}^{t+1} \\ p_{i+2}^{t+1} \end{pmatrix} \tag{3.1}$$

where y_i^t is the experimental data at time t and position i and p_i^t is the prediction at time t and position i.
Since ESN has some memories about the training set, the first 1500 time steps (3 s) are skipped, in order to remove the influence of initial condition from the training set. The testing set starts from time step 7500 (15 s).

3.3 Result

3.3.1 Averaging over 10 pixels

The dataset is generated by averaging over every 10 pixels of y-axis. We first evaluated the RMS error of the prediction in the upper range (y = 0 to 2.6 mm). It contains approximately 100 pixels in y-axis, of the plasma wave, where the movement of the wave is linear.
Fig 3.4 shows one realization of the predicted trajectories. The predictions are close to the testing data and captures the general movement. Other data that have y-value between 0 to 2.60 mm are also tested and show similar results. We have further found out that for $0.05 < \rho < 0.9$, the results are not sensitive to the spectral radius. The root means square errors of 2 s (1000 time steps) of the two predicted time series are less than 0.1. For testing, we did not synchronize any data points. As ESN requires some steps to adapt to the new data, the prediction for the first several time steps are likely to not match the data precisely.

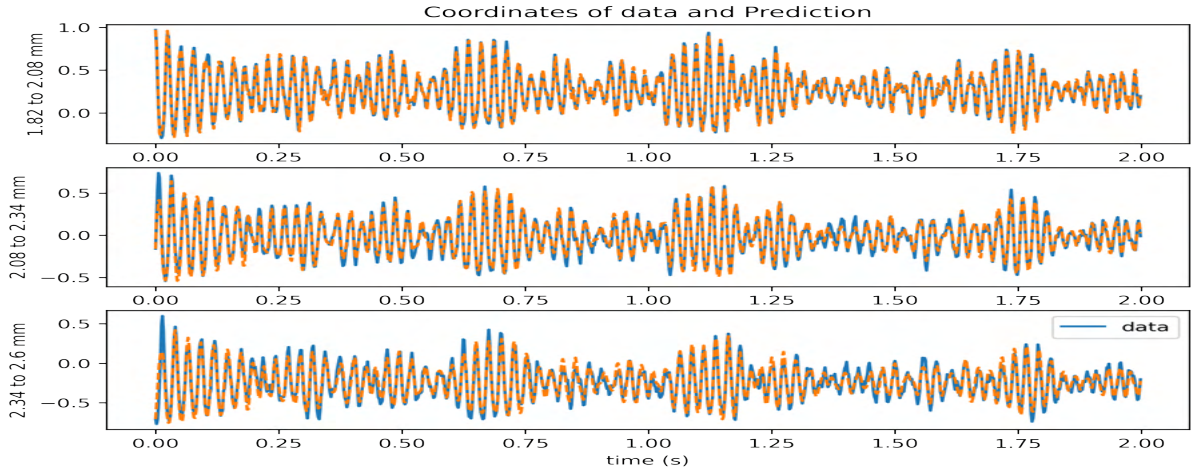


Figure 3.4: Short-term forecast of the intensity at $y = 1.82$ to 2.60 mm. The lines show real data (blue solid line), prediction of ESN (orange dashed line).

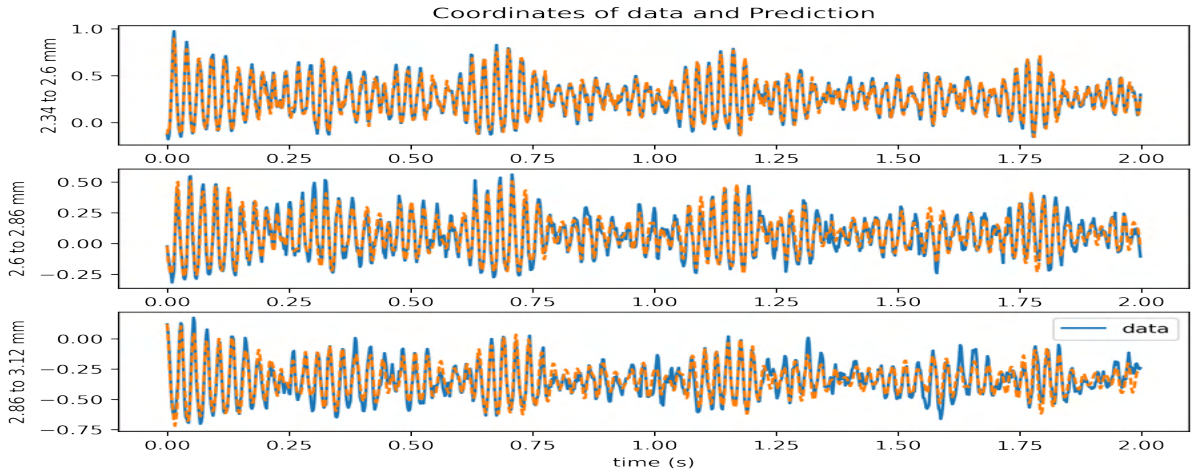


Figure 3.5: Short-term forecast of the intensity at $y = 2.34$ to 3.12 mm. The lines show real data (blue solid line), prediction of ESN (orange dashed line).

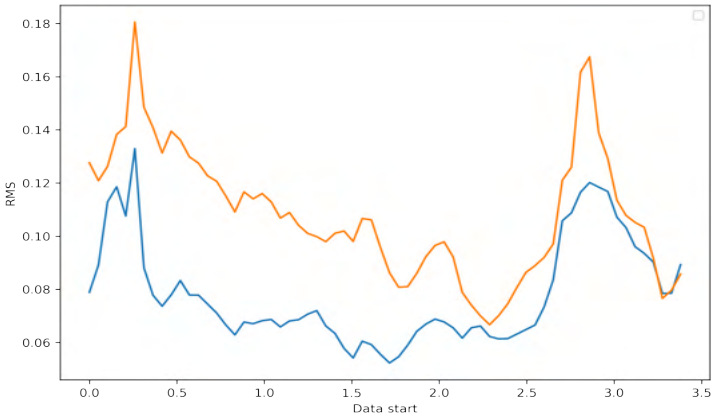


Figure 3.6: RMS errors of the prediction for 1000 time steps in different y-regions. The error of prediction of next 10-20 pixels (blue), 20-30 pixels (orange).

Since the experiment shows that the wave disintegrates at y-value around 2.7 mm, we have also investigated the behavior in this region. Fig 3.5 shows one realization of the prediction. ESNs are set up with the same hyperparameters. The results show that ESNs are still able to capture the wave pattern, but the RMS error is slightly higher than the errors in other regions where the movement is linear.
We have also tested the results at other regions. In Fig 3.6, the RMS errors of the predictions in different regions are demonstrated. The x-axis is the start of the training set. For instance, at $x = 0.0$ on the figure, the training and testing sets are generated from data at $y = 0.0$ to 0.26 mm. The blue curve is the error of the prediction of next 10-20 pixels. For $x = 0.0$, the blue curve is the RMS error of prediction of intensity averaged from $y = 0.26$ to 0.52 mm. The orange curve is for the prediction in $y = 0.52$ to 0.78 mm. Because the formation of plasma wave is not very clear at low values of y, the first several points (data from $y = 0.0$ to 0.5 mm is set to the experimental data) have relatively larger RMS errors. The error continues to decrease, starting from $y = 0.26$ mm until around $y = 2.4$ mm. As already mentioned, the wave disintegrates around $y = 2.7$ mm. The mean value of the wave intensity and the amplitude of the wave vary rapidly among this region. This leads us to believe that it could be the reason that the predictions are less accurate in this region.

3.3.2 Single pixel

In last section, the dataset is generated by averaged over every 10 pixels. The extreme values might be smoothed out, which might lead to loss of information. Especially when the general structure of the wave pattern changes, the averaging might smooth out crucial information and result in a worse performance. Therefore, to further investigate the behavior at region where the wave disintegrates, we adjusted the resolution of training sets to single pixel. In this case, ESNs are trained with high-dimensional data within the region where the wave disintegrated (wave intensity for y-value between 2.34 and 3.12 mm). To make it comparable to the results in last section, the experimental data of the first 10 pixels ($y = 2.34$ to 2.60 mm) is passed into the model and the rest 20 values

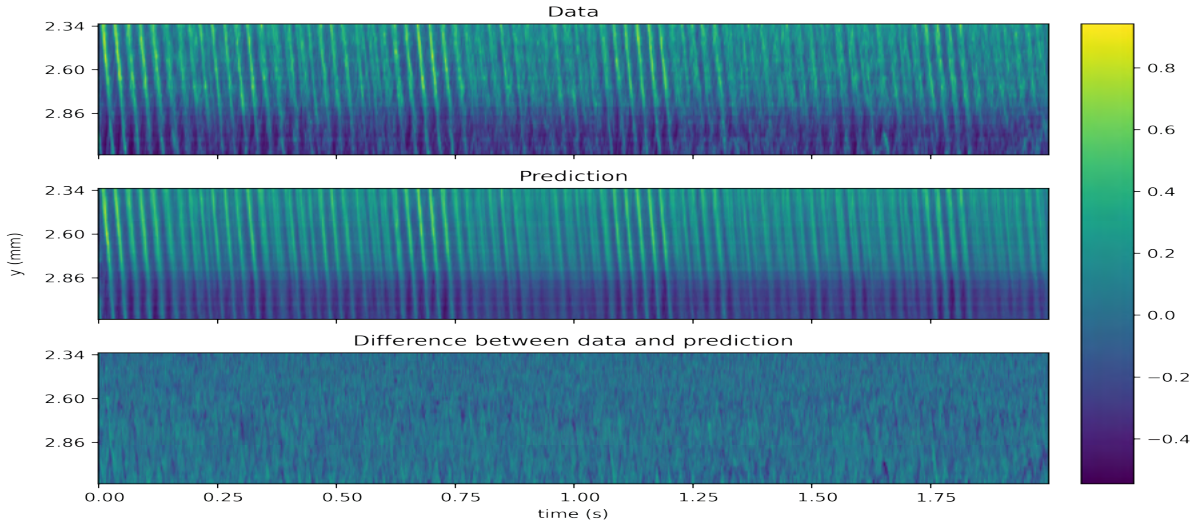


Figure 3.7: Short-term forecast of the intensity at $y = 2.34$ to 2.86 mm at single pixel resolution.

are predicted.

Since the dimension of training and testing sets are increased to 30, it is reasonable to adjust the dimension of the reservoir to a higher number, as more features need to be learnt. The dimension of reservoir is set to 3000 for this section. Other hyperparameters are kept the same as the values shown last section.

Fig 3.7 shows the period grams for the testing data and predictions of intensities from

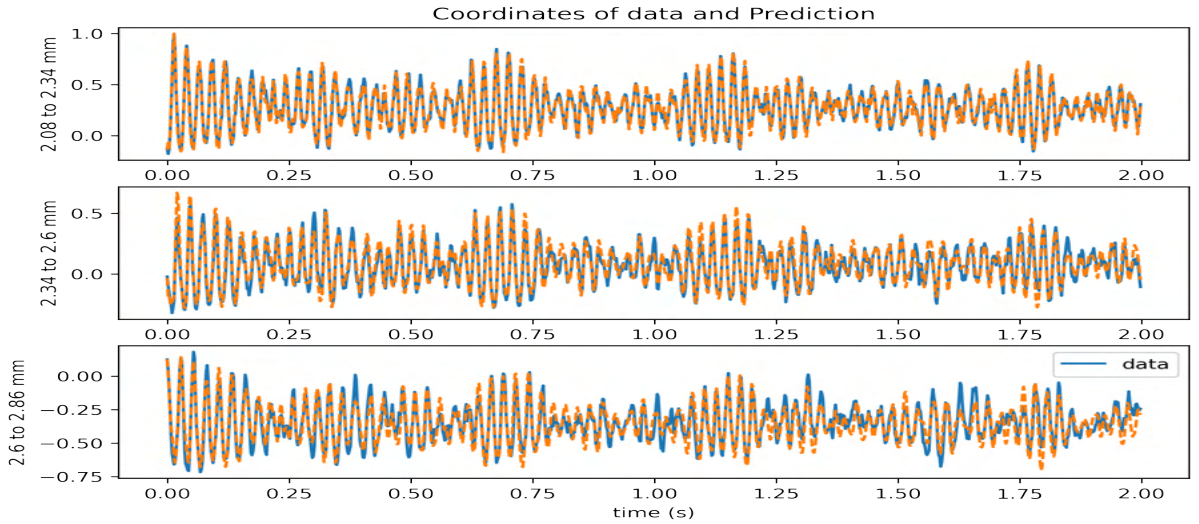


Figure 3.8: Short-term forecast averaged over 10 pixels with the prediction from Fig 3.7. The lines show real data (blue solid line), prediction of ESN (orange dashed line).

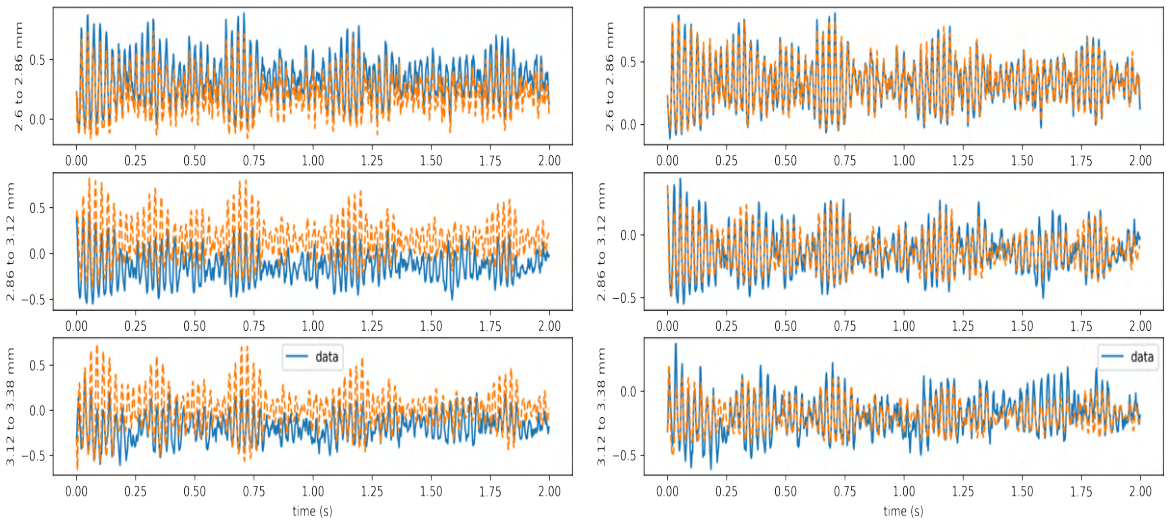
$y = 2.08$ to 2.86 mm and the difference between them. It could be observed that the color of the period grams changes around $y = 2.80$ mm in both the data and prediction, representing a change in the intensities of the wave. The difference between the testing

set and prediction is about the same through the 20 predicted pixels (RMS error < 0.1). Consequently, ESNs are able to predict the movement with the step size of a single pixel. We have also averaged the predictions by every 10 pixels (Fig 3.8). When trained at single pixel resolution, ESNs can capture the peak values slightly better. Since the dimension of the dataset is also larger, the general RMS errors of the predictions are slightly higher.

3.3.3 Cross Prediction

In last section, we showed that ESNs are able to predict the spatial movement of plasma waves. The purpose of this section is to find out whether ESN could predict the movement when the training and testing datasets are from different regions. Essentially, it could show whether the underlying mechanism of wave motion varies at different regions. To evaluate the performance, we forced ESNs to be trained with data from one region on y-axis and tested on another region on y-axis. Because the wave disintegrates around $y = 2.80$ mm, the motion of wave around this region is more likely to be distinct from the others. Therefore, we used data with $y = 2.60$ to 3.38 mm as the testing set. The training sets are generated from other regions of y-values, but with the exact same structure. Both training and testing sets are averaged over every 10 pixels (0.26 mm) and have a dimension of 3. For testing, we used the same mechanism as in section 3.3.1, where we pass in the real data of the first row and the model predicts the intensity of wave at next timestep (Eqn 3.1).

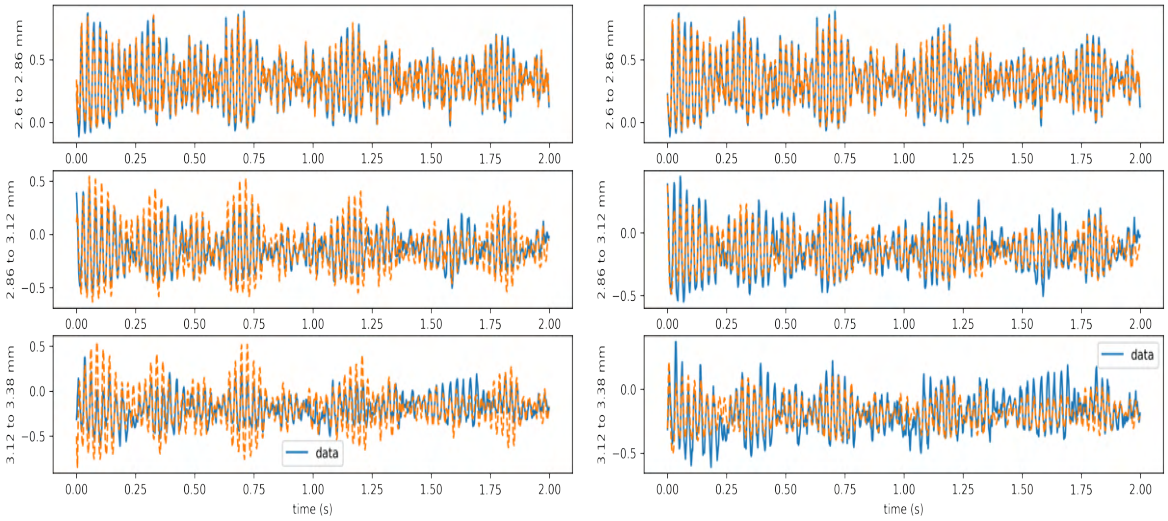
Fig 3.9a and Fig 3.9b demonstrates the results obtained, when the training and testing



(a) (b)

Figure 3.9: Short-term forecast of wave in range $y = 2.60$ to 3.38 mm training data from (a) $y = 1.04$ to 1.82 mm, (b) $y = 2.60$ to 3.38 mm. The lines show real data (blue solid line), prediction (orange dashed lines).

sets are from different regions and the same region. Both forecasts are tested on the same testing set, the only difference is the choice of training data. The predictions with different training and testing sets have errors in the amplitude and the mean intensity



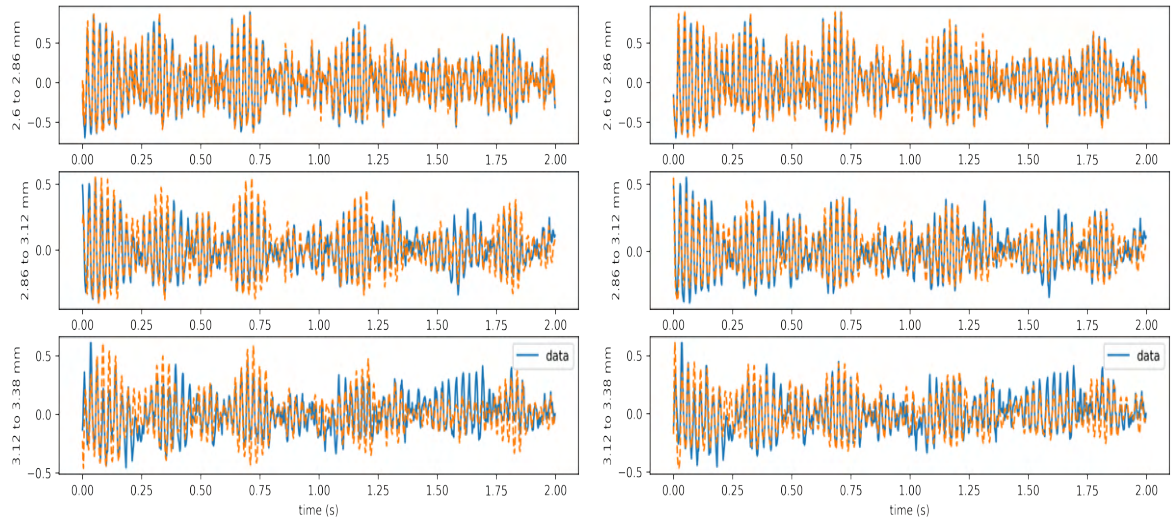
(a) (b)

Figure 3.10: Short-term forecast with mean value corrected in range $y = 2.6$ to 3.38 mm training data from (a) $y = 1.04$ to 1.82 mm, (b) $y = 2.6$ to 3.38 mm. The lines show real data (blue solid line), prediction (orange dashed lines).

(i.e., the wave is shifted up or down) compared to the result obtained, when training and testing uses data from same range. However, the general shape of the forecast is still similar to the testing data.

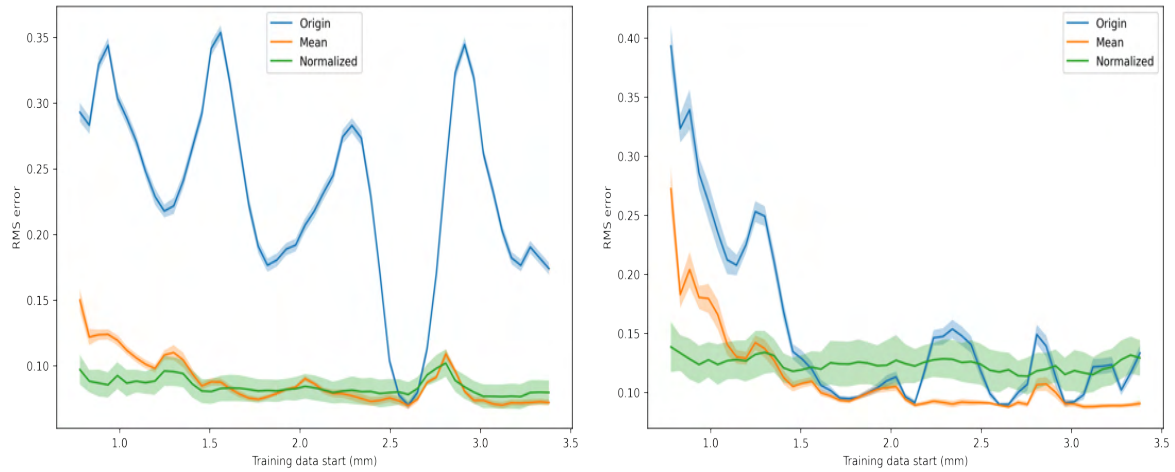
To reduce the error caused by difference in mean intensity and amplitude, first, the mean value of the forecast is adjusted by a constant number, calculated from the difference of the testing set and the original prediction (i.e., shift in the mean intensity). Then, the maximum of the amplitude of the forecast is normalized to the amplitude of the testing sets. The results are shown in Fig 3.10 (after mean correction), and 3.11 (with normalization). After applying the corrections, the results obtained, when the training and testing sets are from the different regions, are very similar to the one obtained, when they are from the same region. To quantify the error caused by the offset of mean intensity and different amplitudes, we calculated the RMS errors of 1000 time steps between testing data and predictions of different training sets. Same as the previous results, we still use data in range $y = 2.6$ to 3.38 mm for testing. Fig 3.12 demonstrates the RMS errors with different training sets. The x-axis shows the starting value of the training set. For example, if the value of x-axis is 0.78, the training set is generated with data in range $y = 0.78$ to 1.56 mm. Each point on Fig 3.12 is separated by 0.052 mm (2 pixels). To compute the results, ESNs are trained 30 times with random initialization, but with the same hyperparameters, and tested on data starting from different time steps. The shaded area represents one standard deviation of RMS error of the predictions.

In general, the original predictions (orange curve) have the largest error. Except at $x = 2.60$ mm, where the training and testing set is from the same region, all three curves have roughly the same errors. After the mean intensity is corrected, the RMS error (blue curve) reduces a lot, which means most of the errors are caused by the difference of mean values. The RMS errors with normalization (green curve) are almost constant



(a) (b)

Figure 3.11: Short-term forecast with normalization in range $y = 2.6$ to 3.38 mm training data from (a) $y = 1.04$ to 1.82 mm, (b) $y = 2.6$ to 3.38 mm. The lines show real data (blue solid line), prediction (orange dashed lines).



(a) $y = 2.86$ to 3.12 mm (b) $y = 3.12$ to 3.38 mm

Figure 3.12: RMS error of 1000 time steps with different training sets in range $y = 0.78$ to 3.9 mm. (a) Prediction of $y = 2.86$ to 3.12 mm (next 10-20 pixels), (b) $y = 3.12$ to 3.38 mm (next 20-30). The lines show error of original prediction (blue), with mean corrected (orange), with normalization (red).

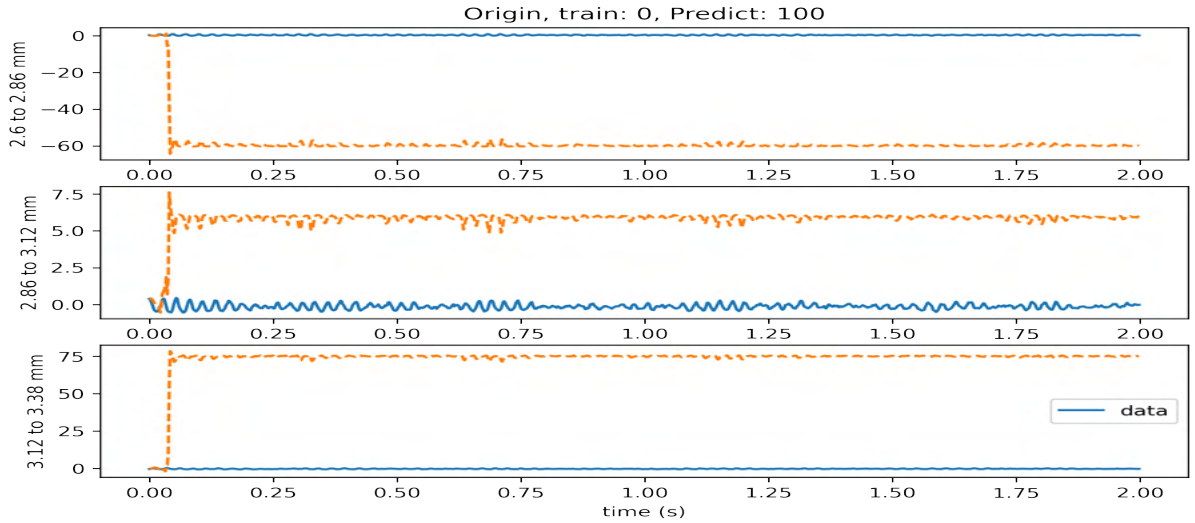


Figure 3.13: Short-term forecast of the intensity at $y = 2.34$ to 3.12 mm with training set from $y = 0.0$ to 0.78 mm. The lines show real data (blue solid line), prediction of ESN (orange dashed line).

when different training sets are used. This leads us to believe that the mechanism between different y-range of plasma wave is similar. Because the normalization is based on the maximum amplitude of the prediction and data, if the extreme value is not captured in the prediction, the error after normalization might be slightly larger than the error with just mean correction. We have also used data with $y = 0$ to 0.78 mm as training set. In this region, the plasma wave has not been formed properly, so the movement is very different from other regions. Unsurprisingly, when the ESN is trained with these data, it fails to make a precise prediction in another region. One example is shown in Fig 3.13.

As shown in the previous section, the mean value of the intensity and amplitude of the wave decrease in vertical direction. This could be the reason that, when the training and testing sets are different, the predictions will fail to capture these features. Therefore, we further compared the difference of the mean intensities of predictions and testing set to the difference between the training set and the testing set. The results are shown in Fig 3.14. The shaded area represents one standard deviation of the values computed from predictions. The x-axis represents the start of y-value of the training set. For example, at 1.0 mm, it uses the training sets from $1.0 < y < 1.26$ mm. And the curve is calculated as the mean of intensity with data in $1.0 < y < 1.26$ mm deducted by the mean intensity with $y = 2.6$ to 2.86 mm. Generally, the curve computed from the predictions is within the error bar, which provides evidence for the existence of correlation. Consequently, if the statistic of the original data is known, the predictions could be easily scaled to the target region, even if the training set is differed from the target.

3.4 Summary

In this chapter, we developed a network with ESN framework and applied it to the spatial prediction of plasma wave. Since the wave propagates downward, the intensities at the

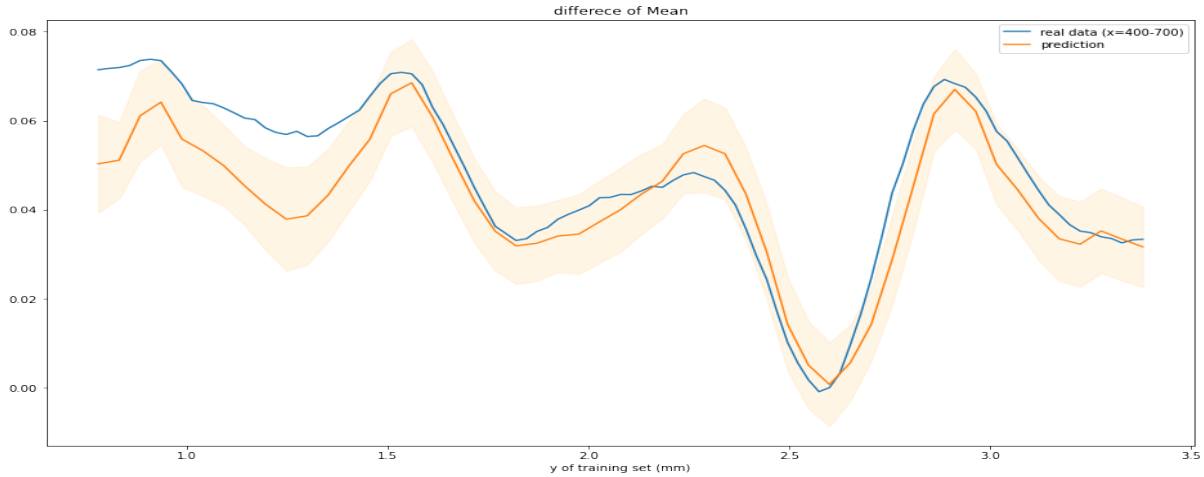


Figure 3.14: Difference of mean intensity from data and prediction of different y-regions to the mean intensity in $y = 2.6$ to 2.86 mm. The lines show real data (blue line), prediction of ESN (orange line).

starting pixel (smallest value on y-axis) are passed into ESNs as the input. Intensities at rest pixels are predicted based on the input.
Firstly, by using different resolutions of datasets (single pixel and averaged over 10 pixels), we found that, in both cases, ESNs can forecast the spatial movement of plasma wave. The accuracy of prediction drops as the wave starts to disintegrate (around y = 2.7 mm). In addition, cross validation is applied to the network. We tried to force the network to predict the movement at the region where wave disintegrates, after being trained with regular data. If the underlying mechanisms are distinct, we expect this prediction to fail, indicating that ESN did not learn anything about the movement at different regions. To accomplish this, we trained ESNs 30 times with training and testing sets with randomly chosen starting time and measured the RMS error of the predictions. For comparison, we also looked at the prediction using training and testing sets from same region. Even though the predictions are generally worse than before and contains error in the mean intensities and amplitude, the general shape of plasmas wave is still captured. After applying a shift to correct mean intensity and normalization of the amplitude, the error drops significantly and almost stays constant when using training sets from different regions. We have further found out that the cause of the error is related to the difference of mean intensities among different regions. As the mean intensity changes along y-axis (Fig 3.3), it leads to an offset in the predicted mean. As a result, with the knowledge about the statistics of the data (i.e., mean intensity and amplitude), ESNs are feasible to make spatial predictions when the training and testing is at different regions. This provides evidence that, even though the pattern disintegrates, the underlying mechanism for spatial movement is still similar.

4 Climate Prediction of Plasma wave

4.1 Motivation and Theory

The focus of this thesis is to make precise predictions of rogue wave (extreme) events of plasma wave. Thus, the reasonable next step is to forecast the long-term behavior of the system. We will investigate the short-term and long-term prediction and optimize the hyperparameters to capture the statistics of the data. In this section, we will use data at $\Delta T = 4K$, 10.4 to 18.2 mm of x-axis. The aim is to make precise prediction for the rogue wave (extreme events). Thus, if the data is averaged over several pixels, the extreme values could be smoothed out. Consequently, we use the signal of a single pixel $y = 1.950$ to 1.976 mm (i.e., the 75th pixel of y-axis).
In this chapter, we will first introduce the basic idea of methods used for data reconstruction (Delay reconstruction and Rolling average). Then, we will present the application and results for the predictions of plasma wave.

4.1.1 Delay reconstruction

Phase space plays an important role in the study of dynamical systems. For deterministic systems, all future states can be determined, when the present state is fixed. Thus, by establishing a point in phase space of the system, the state of the system can be specified. If the system can be modeled mathematically, the phase space is known as the equation of motion. In many cases, the position and velocity. However, in most of the experiments, what we observed are time series (e.g., sequence of scalar measurements) instead of a phase space object. Therefore, we must reconstruct the signals in phase space, which can be solved technically by the method of delay coordinates (also known as lag coordinates).
A delay reconstruction in m dimensions in formed by the vector s_n, given as:

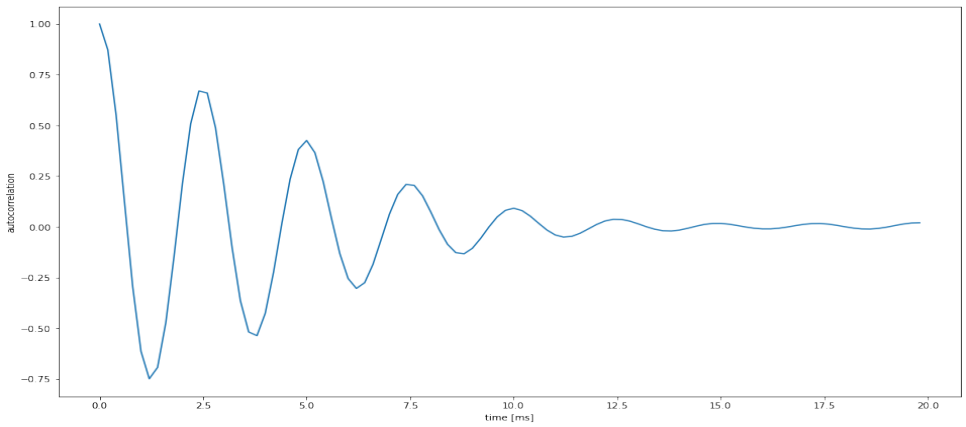


Figure 4.1: Auto-correlation of intensity at 75th pixel on y-axis.

$$s_n = (x_{n-(m-1)\tau}, x_{n-(m-2)\tau}, ..., x_{n-\tau}, x_n) \tag{4.1}$$

where x_n is the measurement at timestep n and τ represents the time lag (in time units, $\tau = \tau\Delta t$).

The next step is to fix the value of dimension m and time lag τ A good estimate of time lag is often difficult to obtain. If τ is too small, the vectors are strongly correlated and clustered around the diagonal. If τ is too large, the elements are nearly independent, so the structure are confine to very small scale. More details about Delay reconstruction could be found in chapter 3 of [12].
In many cases, the first zero of the auto-correlation function yields a good guess. Thus, we computed the auto-correlation function of the signal (Fig 4.1) and found that the wave has a period of around 14 timesteps (2.8 ms). The first zero of the function appears around one quarter of the period, which is about 4 timesteps (0.7 ms). Consequently, time lag of 4 timesteps is used for the experiments in this thesis.

4.1.2 Rolling average

Rolling average, also called "moving average" or "moving mean", is a statistical technique to analyze sequential data by calculating series of mean value over different subsets of the data. In most of the cases, the expected value of the error is around zero. Therefore, by taking mean values, the noise level could be reduced.
In practice, the signal x is averaged over every m timesteps. Mathematically, it is given by:

$$x_n = \frac{(x_{n-m/2} + ... + x_n + ... + x_{n+m/2})}{m} \tag{4.2}$$

where x_n is the signal at time n. Here, we used the center mean method.
Different size of the window is tested to optimize its performance. As shown in Fig 4.2,

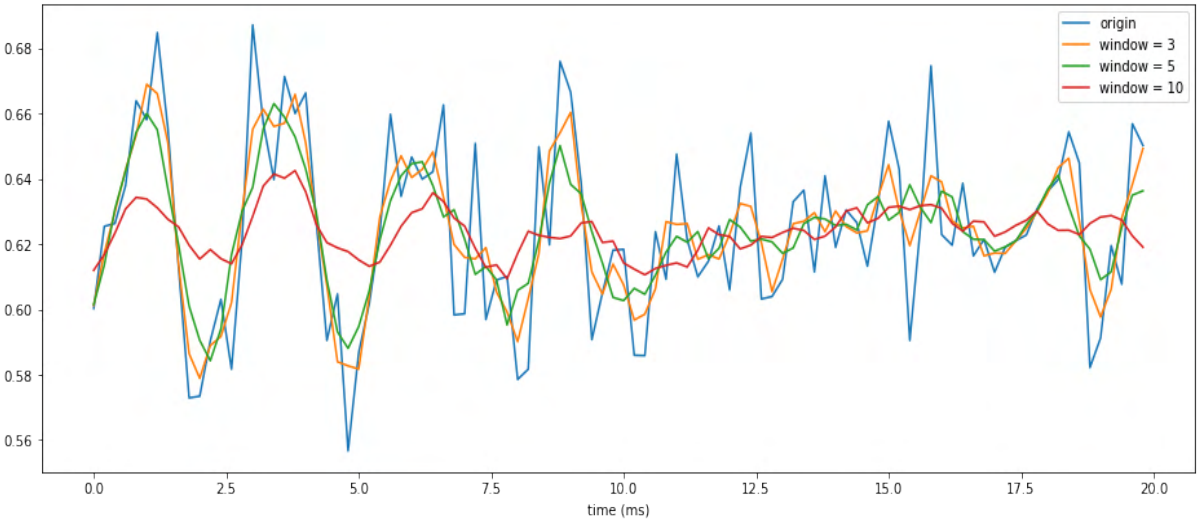


Figure 4.2: Intensity of plasma wave with different window of rolling average.

the original data (blue) contains small fluctuations that last more than 3 frames (0.6 ms), so, with window size of 3, some of the short-term noise could not be filtered out. However, if the window is too large, the important features might also be wiped out. For example, when the window size is 10, the average amplitude (red) is reduced a lot and most of the

extreme values disappear. We have also verified the choice of window size by measuring the Number of peaks versus the height. The peak is defined as the local maximum and the corresponding height is calculated as the difference between the value of the peak and the previous trough. Since we are more interested in the extreme events, we would like the training sets to be denser in larger heights. As shown in Fig 4.3, with window size of

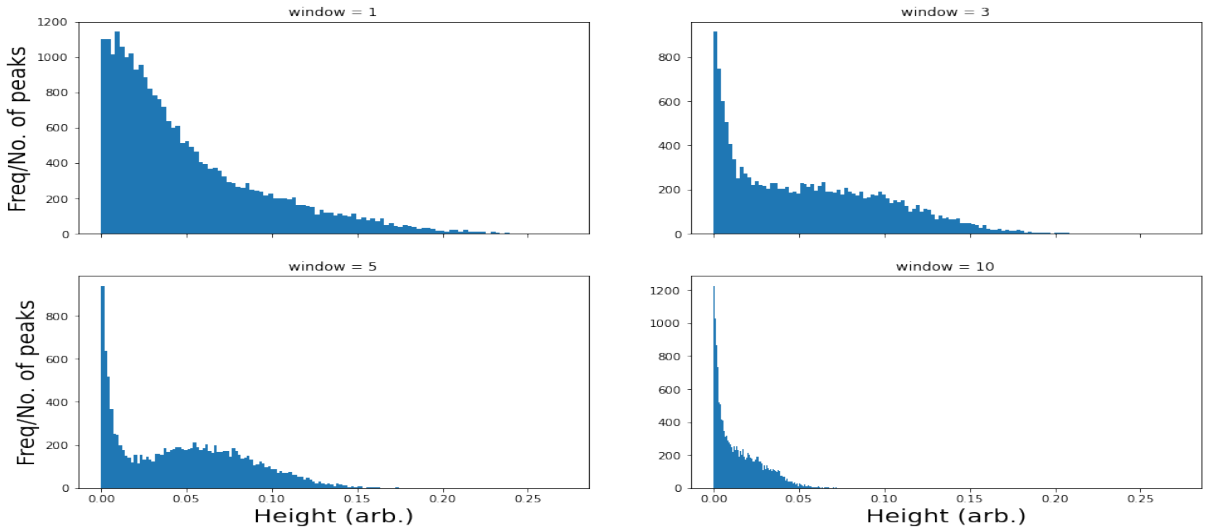
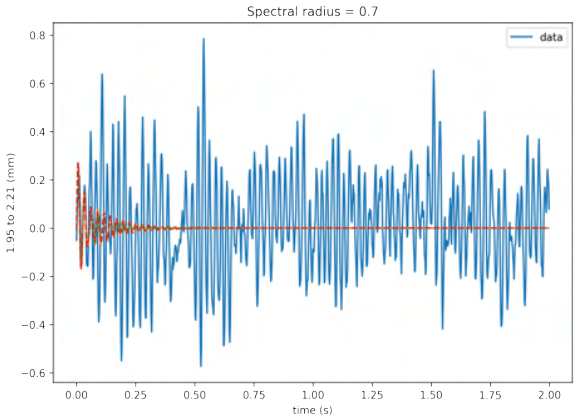


Figure 4.3: Frequency of Number of peaks with different size of window.

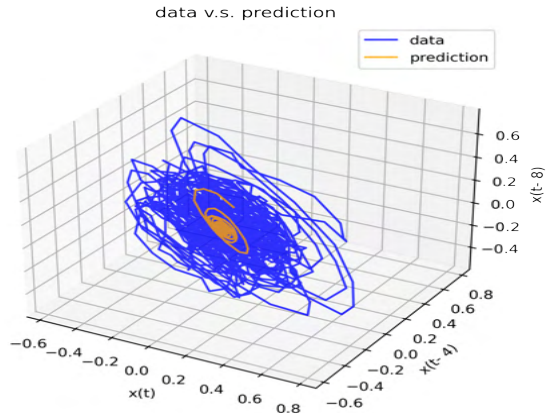
1 (original data), the height distribution is denser around 0, which implies the existence of small fluctuations. When the window size is increased to 3, the height distribution has already been changed. Most of the noise last less than 3 frames are filtered out, which can be observed by the sharp decline of number with height around 0. Yet, after the sharp decline, the number of peaks stays constant as the increase of height, and distribution is still denser in small values. With window size of 5, a second peak could be observed at larger heights. The heavy tail represents that most the short-term fluctuations are smoothed out and the data is more focus on the large waves that reveal the extreme events and general structure of the system. However, if the window is further increased, the distribution becomes similar to the one, when no smoothing is applied. The only difference is the general height level is reduced. In this case, although the noise is filtered, the features of plasma wave, which could provide useful information about the system, are also smoothed out.
Consequently, we applied the rolling average with window size of 5 to generate the training and testing sets for our experiments. This is also useful for the measurement and forecast of rogue wave, which will be explain in detail in next chapter.

4.2 Short-term Prediction

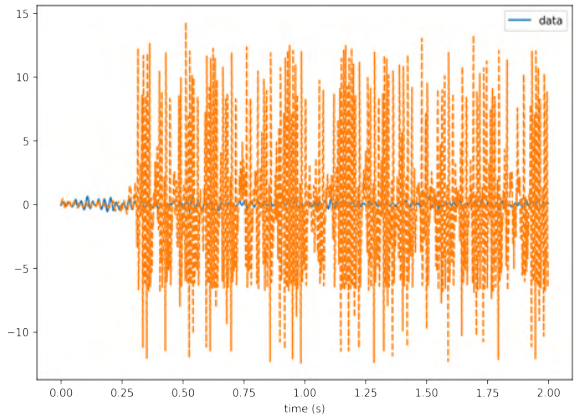
Before the prediction of climate and long-term behavior of plasma wave, we first looked at the predicted trajectories. The training and testing sets are normalized as in chapter 3. Hyperparameter optimization is always a crucial step of machine learning techniques. Different choice of hyperparameters will affect the performance and efficiency of the model.



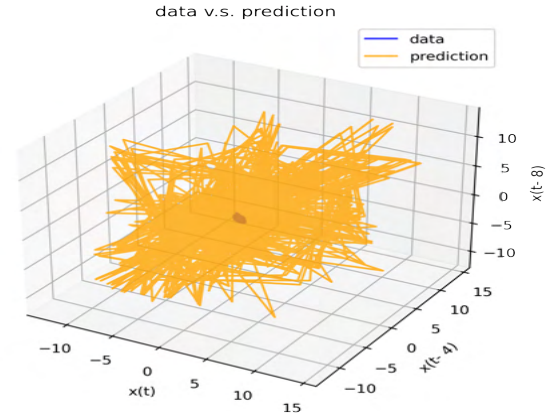
(a) $\rho = 0.7$



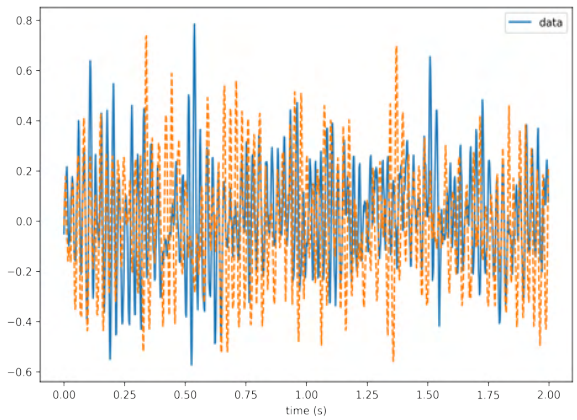
(b)



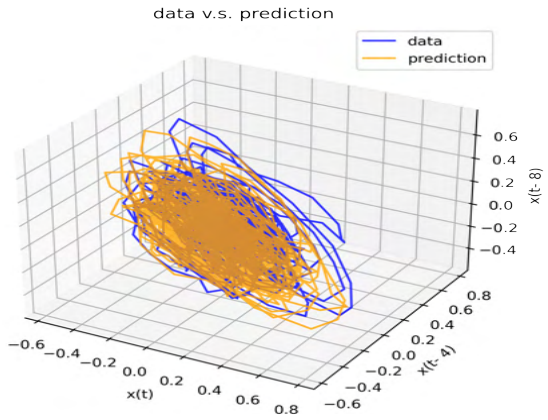
(c) $\rho = 0.9$



(d)



(e) $\rho = 1.3$



(f)

Figure 4.4: Short-term forecast with different spectral radius. The lines show real data (blue solid line), two realizations of prediction of ESN (dashed lines).
(a),(c),(e) demonstrate the predicted trajectories of intensity. (b),(d),(f) demonstrate the 3D representation.

The spectral radius ρ is the important one for reservoir computing. It is believed to be related to the memory of the reservoir. Larger ρ often leads to a longer memory of the time series and more chaotic forecasts.
In this section, the reservoir is set to a dimension of 1500, and we vary the value of ρ to investigate the behavior. ESNs are trained with three-dimensional data with 6000 time frame (3 s) and tested with data starting 1500 time frames (0.75 s) after the end of training set. Since the delay coordinates are used, the inputs also contain information from the past frames. For $n < m\tau$, because part of the input needed is earlier than the start of prediction, these values are directly chosen from the dataset. Otherwise, the inputs are based on latest predictions. i.e., only the first value of the output, which represents the current frame, is recorded and used for later predictions. The governing equation is shown below,

$$\begin{pmatrix} y_0(n) \\ y_0(n-\tau) \\ \cdots \\ y_0(n-m\tau) \end{pmatrix} \rightarrow RC \rightarrow \begin{pmatrix} y_0(n+1) \\ y_1(n+1) \\ \cdots \\ y_m(n+1) \end{pmatrix} \tag{4.3}$$

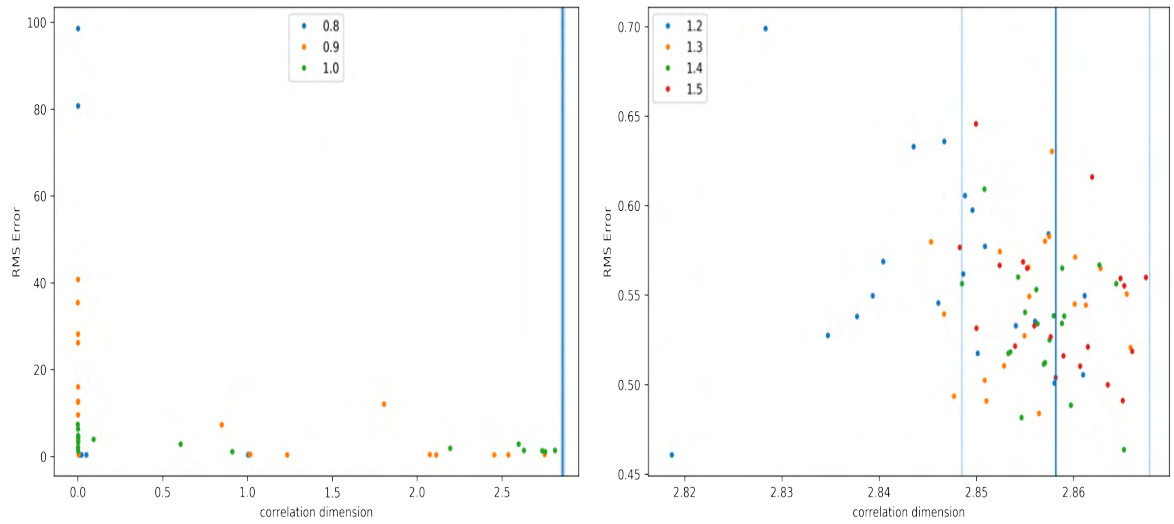
where $y_i(n)$ are the output of the ith row at frame n and τ is 4 time frames.
In Fig 4.4a, the predictions with $\rho = 0.7$ is demonstrated. We could observe that the amplitude of prediction damps away very quickly (within 0.2 s) and the prediction becomes a constant function. In other realizations with $\rho = 0.7$, the predicted wave might also become simple periodic wave. The 3-dimensional plot is generated by Delay reconstruction. We used values of $y_0(n), y_0(n-4), y_0(n-8)$ as the one shown in eqn 4.3. In the 3D-reconstruction, the trajectory also converges into a fixed point at later time. Other values of $\rho < 0.8$ is also tested, all the predictions share similar behaviors.
When spectral radius is increased, we found that, around $\rho = 0.9$, the predictions become extremely unstable. In Fig 4.4c, it shows one realization with $\rho = 0.9$. At the beginning, the predicted wave is close to the data. However, the amplitude of the prediction deviates at around 0.25 ms and becomes much larger than the experimental data. Although most of the predictions are unstable, we have also observed some realizations, about 10%, that gives a reasonable prediction and have similar structure as the testing sets. The details about this phenomenon will be given in the next section. For $\rho > 1.3$, the predictions become stable again. As shown in Fig 4.4e, the average amplitude and frequency of the prediction match the experimental data. In the 3D representation, the attractors have similar shape between the prediction and testing sets.

4.3 Long-term Prediction

The same settings of the ESN and dataset from last section are used. To investigate the long-term behavior of the prediction, we first computed the correlation dimension and maximum Lyapunov exponents for the experimental data. For these calculations, we used 20000 time frames of the dataset (40 s), and averaged over 100 regions starting at different initial frames of the data. In order to optimize the spectral radius ρ on the predicted climate, ESNs are trained 20 times for each choice of ρ with other hyperparameters kept the same, but random initializations of reservoir and input matrix W_{in}.
The results are shown in Table 2. The first row is computed from the experimental data.

ρ	RMS error	Correlation dimension	mle
Data	-	2.8582	0.0017
0.5	0.358	0.0037	-
0.8	9.314	0.5094	-
1.0	2.960	0.8672	-
1.4	0.533	2.8570	0.0001
1.7	0.541	2.8578	0.0005
2.0	0.538	2.8578	0.0007
2.3	0.537	2.8591	-0.0002

Table 2: Mean value of RMS error, correlation dimension and maximum Lyapunov exponents over 20 realizations of ESNs with different spectral radius.



(a) $\rho = 0.8$ to 1.0 (b) $\rho = 1.2$ to 1.5

Figure 4.5: RMS error and correlation dimension of the predict wave with different spectral radius. The vertical line represents the correlation dimension of the data.

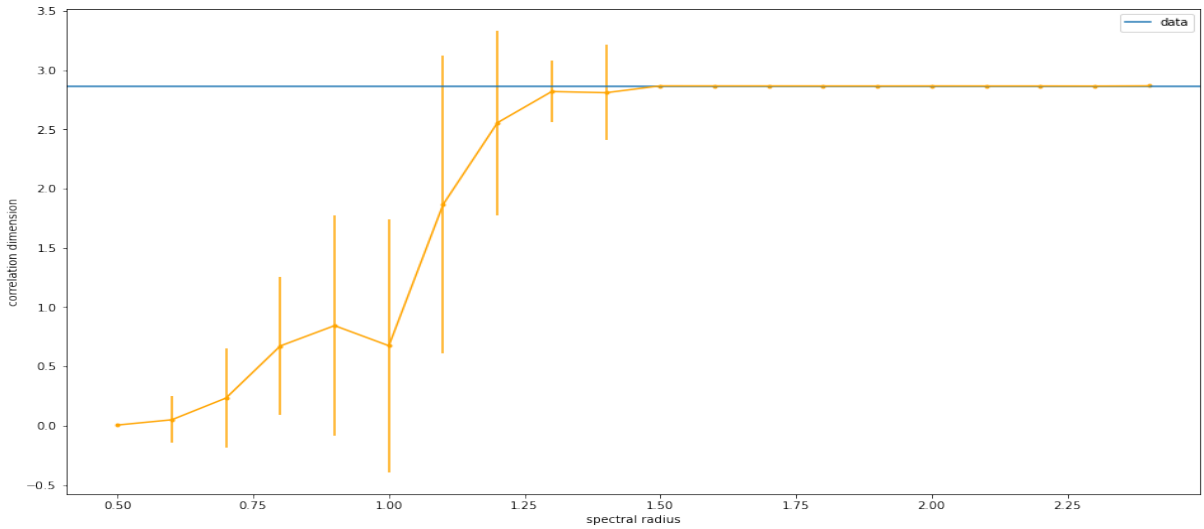


Figure 4.6: Mean value of predicted correlation dimension with different spectral radius. The blue line shows the value measured from experimental data. The orange line shows the values from forecast trajectories with error bar.

To measure the correlation dimension and Maximum Lyapunov Exponent for predictions, the outputs are adjusted to 3-dimensional by using delay reconstruction. From these results, a shift of the statistics of predictions could be observed. With small values of spectral radius ($\rho = 0.5$), the correlation dimension is around 0, which is a lot smaller than the value for experimental data (2.86). When the reservoir has $\rho = 0.8$ to 1.0, the predicted amplitudes are much larger than the amplitude of the experimental data, which leads to a tremendous increase of RMS error. One example is shown in Fig 4.4c. The highest RMS errors occurs when $\rho = 0.9$ and leads to highly unstable results.
The predicted correlation dimension also varies around $\rho = 0.8$ to 1.0. With $\rho < 0.8$, the most of the predicted correlation dimensions are around 0, which means that the prediction either becomes a fixed function or a periodic wave (Fig 4.4a) after several frames. Around $\rho = 0.8$, some of the predictions start to have similar structure as the data. However, there are still many predictions that fails to capture the long-term statistics. To find out exactly how well the predictions are, we simulated 100 realizations with randomly chosen initial conditions and found that, with $\rho = 0.8$ about 20% of predictions captures the climate, as shown in Fig 4.5a. When the spectral radius is further increased, the forecasts become more stable and could mostly capture the structure with $\rho > 1.3$. The predicted correlation dimension keeps around 2.858, which is very close to the real value, as shown in Fig 4.5b.
The Maximum Lyapunov Exponent cannot be measured properly with $\rho < 1.0$, since the predictions are either unstable (i.e., amplitude is too large), or decay into a constant function shortly. Because the data are from real system (plasma wave), the Maximum Lyapunov Exponent cannot be positive. This is also shown in Table 2, as all the measured values are around 0.

4.4 Summary

In this chapter, we first smoothed the time series with moving average to filter out the high-frequency noise. Then, ESNs are trained and tested on the reconstructed datasets. We varied the spectral radius of the reservoir to optimize the forecast. As mentioned in the section 2.3, $\rho < 1$ is often considered as a necessary and/or sufficient condition for ESN. Yet, in our experiment, we have found that, with $\rho > 1$, ESNs have better performance in predicting climate of plasma waves.

Fig 4.6 demonstrates the results obtained with different spectral radius. From the diagram, a clear shift in the correlation dimension could be observed. With $\rho = 0.5$, all the forecasts become a periodic wave or decay into a constant function shortly, which leads to correlation dimension of 0. As the increment of ρ, the average value of predicted correlation dimension starts to increase and the error bar (i.e., the standard deviation) becomes larger. The error bar represents the distribution of predicted correlation dimensions. Small error bar means most of the predictions have similar statistics. On the other hand, if the error bar is large, it means that the predictions are unstable and have distinct behaviors. With ρ in range 0.8 to 1.2, the predictions are unstable and trends to have much larger amplitudes than the original data. However, some realizations ($\sim 20\%$) could capture the long-term statistics. This could be observed as the increment of correlation dimension and the error bar. As ρ is further increased, the predictions become more stable and after $\rho > 1.3$, all predictions capture the structure with small variation, as the predicted correlation dimension is very close to the experimental data.

As a result, with a proper choice of spectral radius ($\rho > 1.3$), ESNs are feasible to reproduce the climate of plasma wave.

5 Rogue Wave Prediction

In section 4, we showed that with proper setting of hyperparameters, ESNs can forecast the long-term behavior of the plasma waves. In this section, we focused on the short-term predictions of extreme events. Here, we chose rogue wave as the target. First, we will briefly introduce the concept and definition of the rogue wave events in complex plasmas, with the statistics used to determine the performance of predictions. Then, the performances of short-term forecasts of rogue waves are presented.

5.1 Rogue wave

As mentioned in the introduction, rogue waves were first introduced in Oceanography [23]. These events have large crest-to-trough height exceeding two times significant height (Hs), which is defined as the mean of largest third of wave heights. Rogue waves in complex plasmas [22] were first reported experimentally by Tsai et al. [32]. They are defined with similar concept as in the Oceanography. The amplitude of the wave height in complex plasmas is measured as the maximal micro-particle number density in the wave crest, which in turn can be approximated with the increase in average pixel brightness [29]. Lin and I [13] demonstrated that the formation of rogue waves in complex plasmas is associated with the synchronization of 3D particle focusing by preceding distorted waveforms at different scales.
Here, the definition of significant height is the same as shown before. The threshold for rogue waves is set to 1.3∗Hs. If the threshold is 2∗Hs, only 2 rogue wave events occur within 300,000 frames (600 s), which is not enough for training and testing. However, with threshold of 1.3∗Hs, much more events appear (about 1300 events within measurement). Some sample rogue events are shown in Fig 5.1. Please notice that the plotted dash-dotted

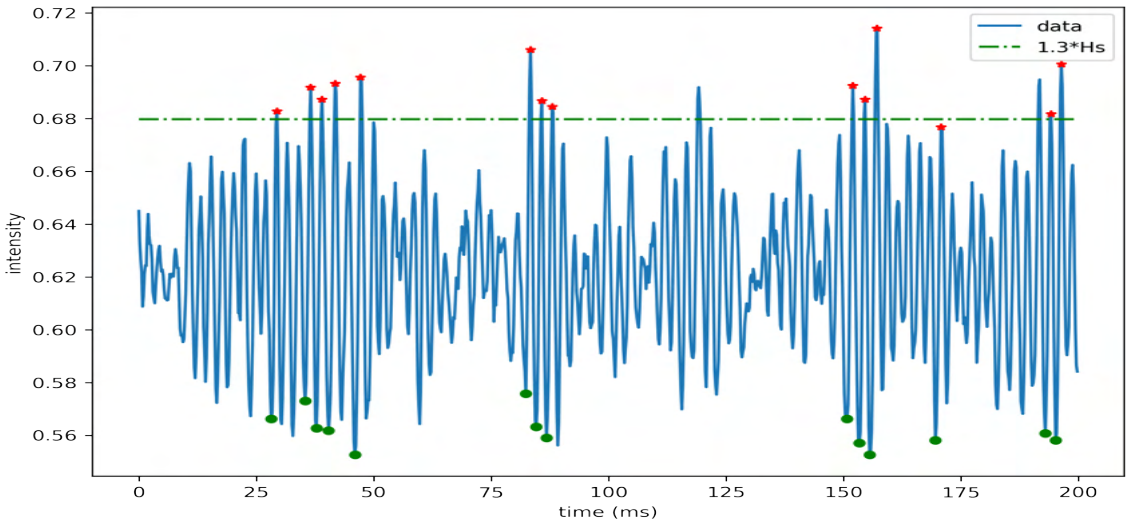


Figure 5.1: Sample image for rogue wave events. Red stars are peaks of the event and green dots for troughs. The dash-dotted line shows one-half of significant height plotted from mean intensity.

line is one-half of Hs. Since the height is measured as difference between trough-to-peak,

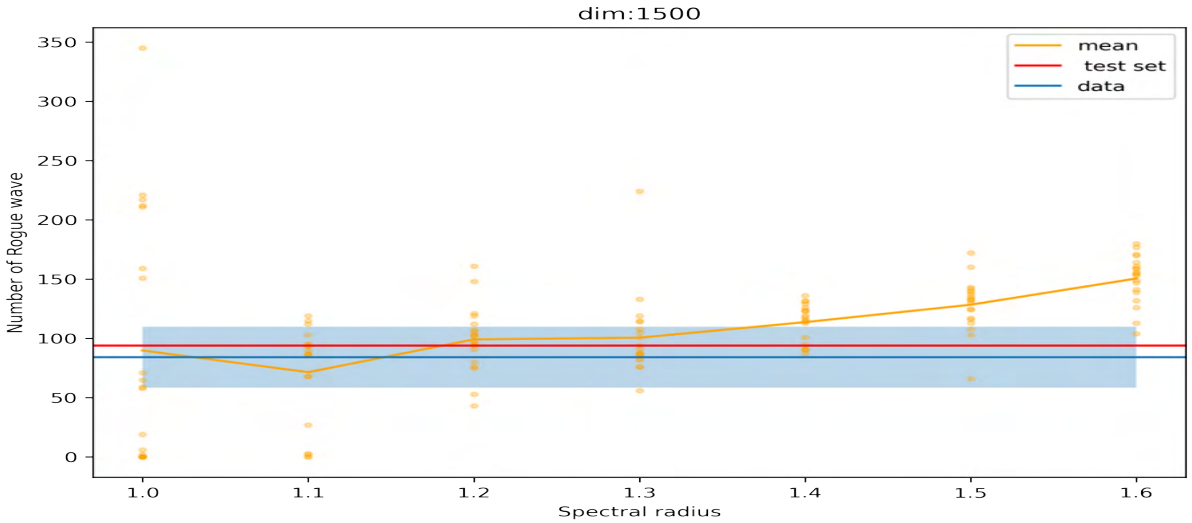


Figure 5.2: Number of Rogue wave events in 20000 frames of prediction. The lines show real data (blue), testing set (red) and prediction (orange) with the values of each realization (orange dot).

so even if the peak value is higher than this line, it might still not be recognized as a rogue wave.

5.2 Statistics

Before the actual predictions of rogue waves, we first examined the performance of ESNs in capturing long-term behavior. As shown in last chapter, ESNs can reproduce general climate of plasma wave. Here, we focused on the statistics of rogue wave events. We use the significant height, which is used to determine rogue wave, and the average number of rogue waves within 20000 frames, which shows the frequency of occurrence, as the reference.

To optimize hyperparameters of the reservoir, we varied spectral radius and measured the number and significant height of rogue waves within 20000 frames of prediction and compared them to the testing sets.

The results are shown in Fig 5.2 and 5.3. The blue line is the mean value of 20000 frames of data with distinct initials. The shaped area shows three standard deviations of these measurements. And red line represents number and significant height of rogue waves in testing set.

For $\rho < 1.0$, ESNs are not able to capture the general statistics of the system. The predictions completely fail and contain no rogue wave event. Thus, the figure only demonstrates results with $\rho \geq 1$.

At $\rho = 1.0$, although the mean value of predicted rogue waves is close to the value measured from experimental data, it contains large variance. Many realizations do not contain any rogue waves in the prediction, while the others contain more than 300 events within 20000 frames (only about 100 events occur in experimental data). Similar behavior is observed with $\rho < 1.2$.

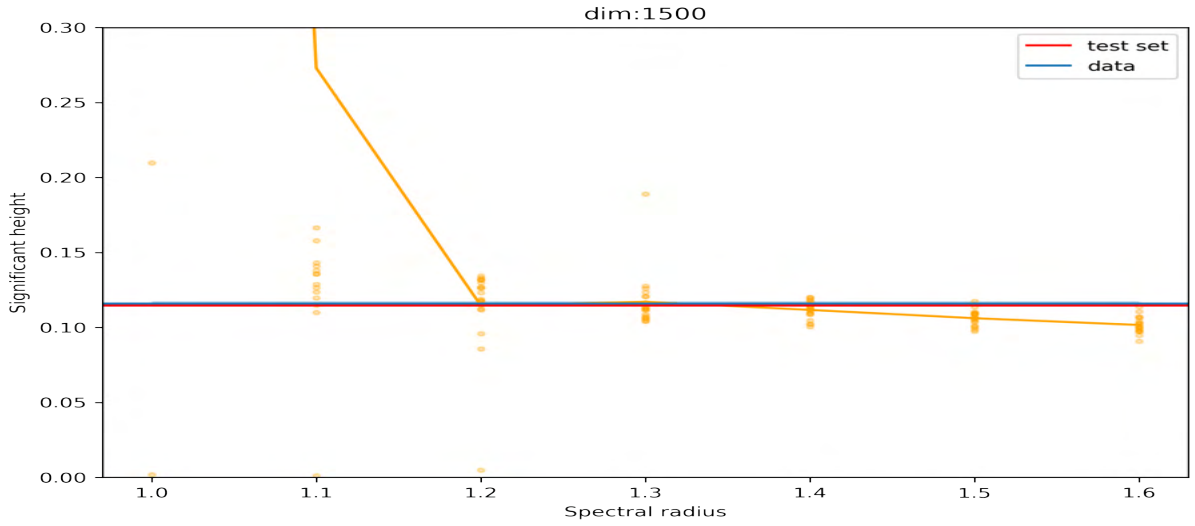


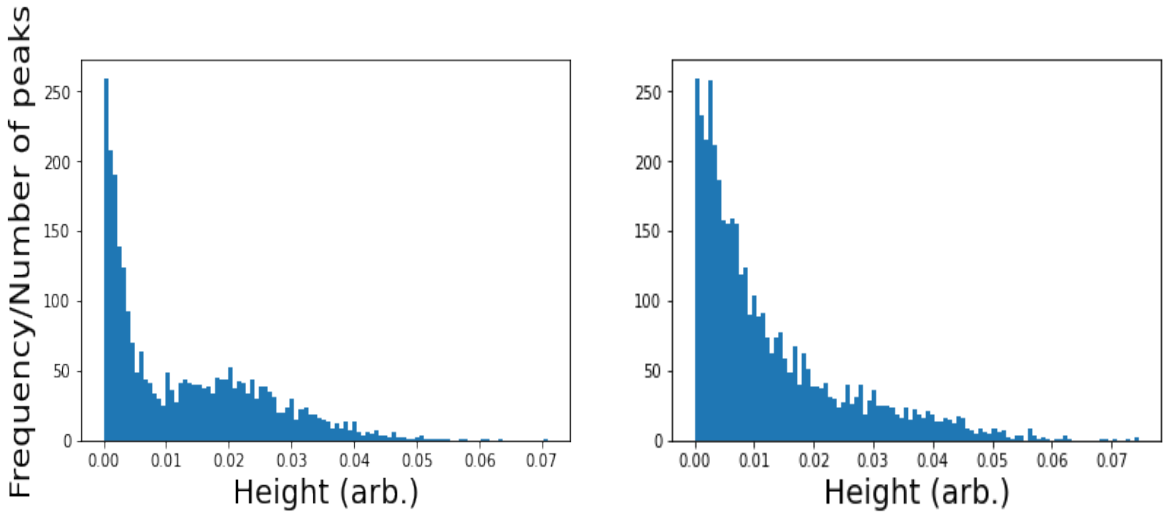
Figure 5.3: Significant height computed from 20000 frames of prediction. The lines show real data (blue), testing set (red) and prediction (orange) with the values of each realization (orange dot).

Around $\rho = 1.2$, the average number of predicted rogue wave is similar to the experimental data and the predicted number continues to increase as the increments of ρ. Meanwhile, the predicted significant height is closed to the data and continue to decline as the increase of ρ. We have also checked the amplitude of the predicted time series. However, the average height of the prediction is almost same at different value of ρ. The predicted height distributions are presented in Fig 5.4. Both distributions have similar tail at larger heights. With $\rho = 1.3$, it is similar to the distribution computed from the data, where a second peak could be observed. The height distributions with different windows size are shown in Fig 4.3. However, with $\rho = 1.6$, the heights are denser around 0, which means there are small fluctuations in the prediction, that do not occur in the training data. Since significant height is computed from $\frac{1}{3}$ of the maximum heights, if the height distribution is denser around 0, the mean value of the $\frac{1}{3}$ of the maximum heights declines. Consequently, the significant height also declines as the increase of spectral radius.

Since the goal is to make precise short-term prediction of rogue wave events, it is not ideal if the prediction contains much more events than the experimental data, as it might give many false predictions when no rogue wave occurs. As a result, $\rho = 1.3$ was chosen, as it best captures the frequency of occurrences.

5.3 Prediction

For short-term prediction, the testing sets are generated a few steps before the peak of rogue wave event with different choices of rogue wave, but with the constraint that there is no overlap with testing sets. ESNs forecast the next 10000 time frames. As significant height (Hs) is related to the general amplitude and statistic of the wave intensities, we computed Hs of the prediction based on the predicted values and used it to classify rogue waves in the forecasts. Then, we check whether ESNs could capture the events (i.e., the



(a) $\rho = 1.3$ (b) $\rho = 1.6$

Figure 5.4: Frequency/ Number of peaks corresponding to different height with spectral radius of 1.3 and 1.6.

prediction also contains rogue wave events at the same time frames as testing set). Since measurements are discrete, peaks might occur between two time frames. If the predicted peak only differs by one frame, we still consider it as a successful capture. Here, we focused on prediction of when the rogue wave occurs, so, even if the height of the predicted wave is not an exact match of testing set, as long as the predicted peak appears on the same time frame as the testing data, it is classified as a successful capture.
Rogue wave events are likely to occur continually, so we also checked the captures of the second and third event. Normally, there are no more than 4 continues rogue waves, so, we only counted the correct captures within the next 50 frames (about 3-4 full waves).
We also found that, by increasing the dimension of input, ESNs perform better in capturing the extreme events. Although larger dimension does not have significant influence on long-term behavior, accuracy of short-term captures of rogue waves, especially within the first 50 frames of predictions, is increased. The lag constant is still set to 4, as in section 4.1.1. The dimension of the inputs is increased to 8, so that the values within two wavelengths before the current frame could be provided to the reservoir.

5.3.1 Correct predictions

The percentage of successful captures are shown in Fig 5.5. Notice that for prediction within 7-8 steps before the peak of rogue wave, the first capture is the second event, which is about 14-21 frames after the prediction starts. Since the height of rogue wave is computed as the difference between the peak and the trough before it and troughs are generally 7-8 steps before peaks, for these predictions, the first rogue wave has already begun before the prediction starts.
From the results, we found that, when the start of the prediction is around the crest value of rogue waves, ESNs could capture the next peak with high accuracy. For example, if

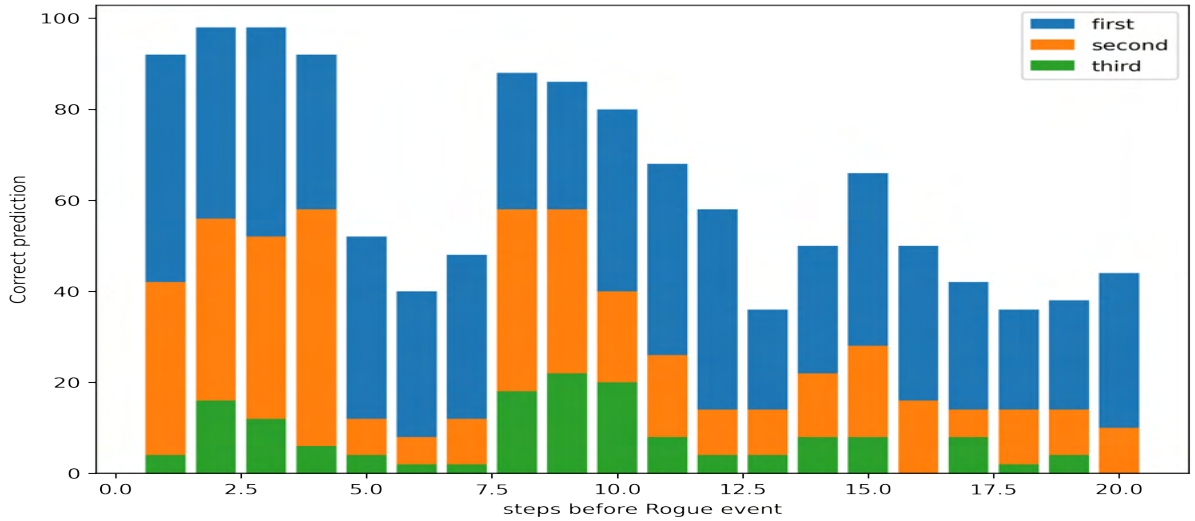


Figure 5.5: Histogram of percentage of correctly identified rogue wave events. This percentage is calculated using 50 realizations with different starting times for the prediction.

the start is around 8 frames before the peak value, more than 80% of the predictions managed to capture the next event and 60% of them capture the next two events. One example is shown in Fig 5.6, where it successfully captures the events.
As shown in the histogram, the ratio of successful predictions oscillates as the prediction starts at different time. More events could be captured when prediction starts at 8 and 15 frames before the rogue wave. If the initial is around the mean intensity (zero crossing), the accuracy drops significantly. We have further investigated this phenomenon around these regions and found out that one of the reasons is that predictions go into the wrong direction. For example, the intensity of the testing set increases, but the intensity of prediction declines. One of the explanations might be that the data is almost symmetric around the mean intensity. If our algorithm also starts forecast around the mean intensity, probability of intensity to increase or decrease is almost the same. On the other hand, if forecast starts at peak values, the intensity would be likely to declines, which is easier for ESNs to capture the trend.

The next step is to further break the symmetry. There are several ways for symmetry breaking. For instance, by introducing extra terms with higher order or derivatives. Here, we first tried a simple way, by adding more dimensions to the input of ESNs (i.e., increase the dimension of training and testing sets). To make the result comparable, training set still take data within 2 periods. As the previous results are obtained with datasets of dimension of 8 and lag constant of 4, the product, which represents the earliest value included in input, is from 28 frames before the current frame. We increased the dimension to 14, so lag constant is reduced to 2 frames. The earliest value in the input is 26 frames, which is still within 2 periods.
Reservoirs with same dimension and setting of hyperparameters are used for comparison. The result is shown in Fig 5.7. With higher dimensional datasets, ESNs perform better around zero crossing (i.e., around 5-7 frames) and become more stable. Unlike before with smaller dimension, the ratio of correct prediction is much higher, when the predic-

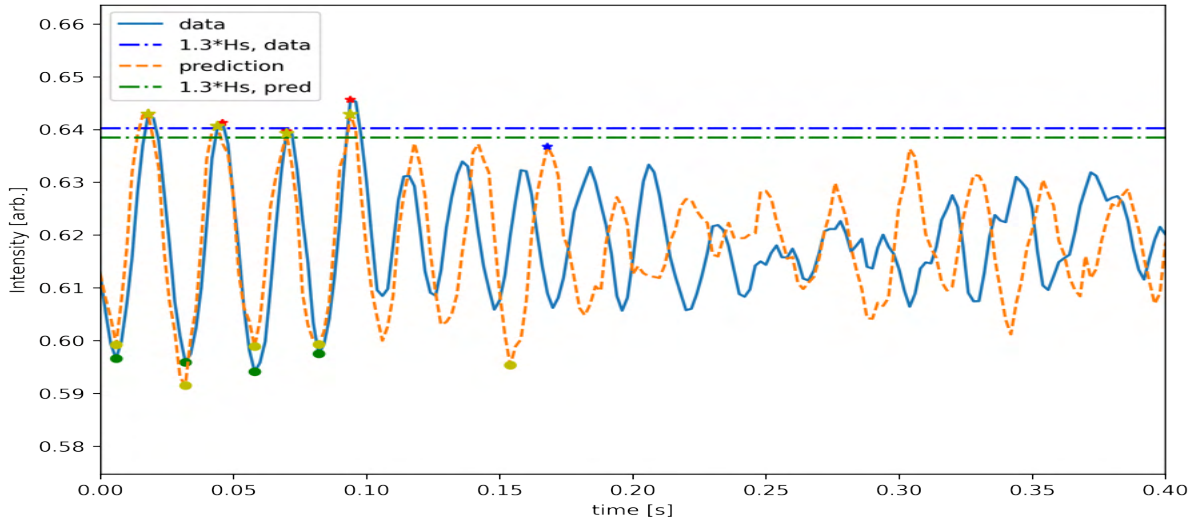


Figure 5.6: Intensity of the prediction and testing data testing set (blue), prediction (orange dashed line) and the significant height for testing set (blue dash-dotted), prediction (green dash-dotted). The algorithm begins to predict at 10 frames (20 ms) before the peak of the rogue wave event.

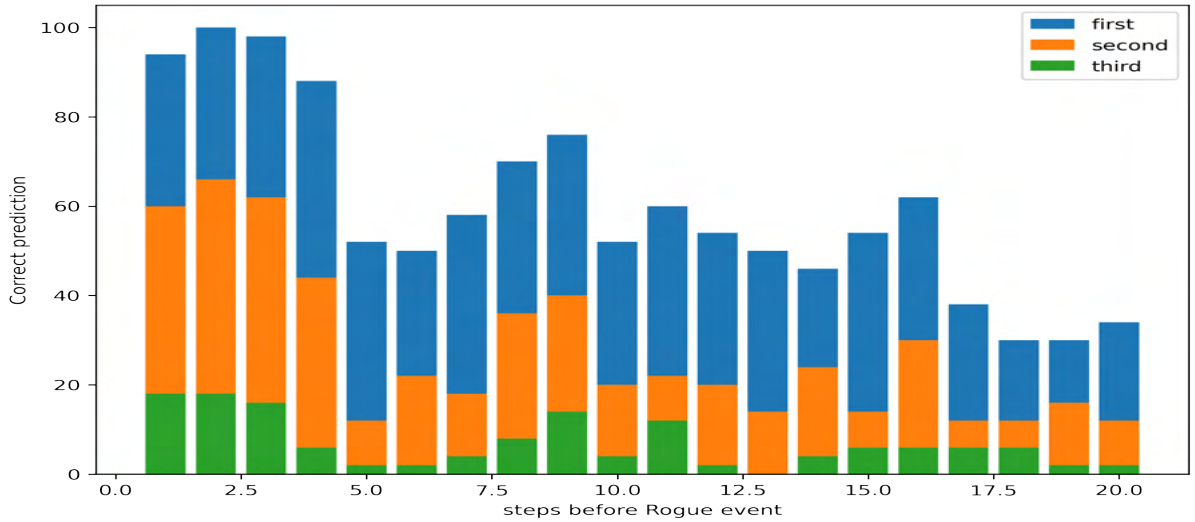


Figure 5.7: Histogram of percentage of correctly identified rogue wave events with training set of 14 dimensions and lag constant of 2 frames. This percentage is calculated using 50 realizations with different starting times for the prediction.

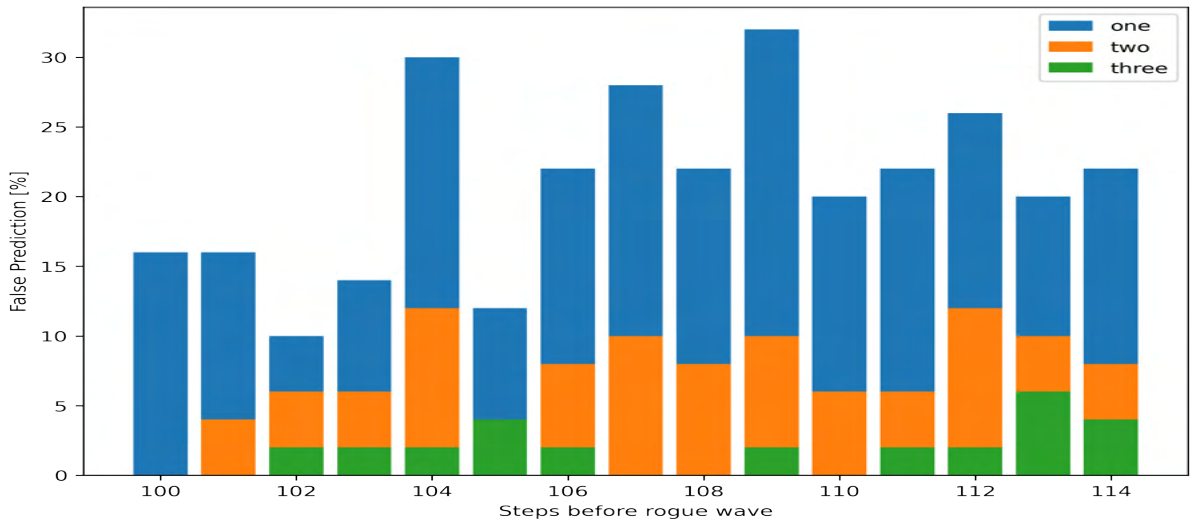


Figure 5.8: Histogram of percentage of false predicted rogue wave events. This percentage is calculated using 50 realizations with different starting times for the prediction.

tion starts at crest value. Here, the difference between different starts is much smaller. However, the ratio of successful captures at crest values declines slightly. Since the dimension of input has increased, there are more features need to be learned. Some important features which were learned could be omitted, as the increment of dimension, which could leads to less accurate prediction.

5.3.2 False positive

In last section, we showed that ESNs could capture the rogue wave event within a short period of time. We have also tested the results on data that does not contain any rogue wave event to check for false positives. As shown before, the initial point of the forecast plays an important role in the performance. In order to better understand the influence when no rogue wave occurs, the predictions also start at different time within one period. The testing sets are chosen such that no rogue wave occurs in the first 50 frames. For simplicity, the testing sets are generated from 100-114 frames before some rogue events occur. The network takes the input with dimension of 8, which is equivalent to the input used for Fig 5.5.
In Fig 5.8, it shows the percentage of false predictions within the first 50 frames. On average, we observed about $20 - 25\%$ of the prediction that contains at least one false prediction, which is similar to the expected number of rogue waves in 50 frames (i.e., on average, there is one rogue event in every 200 frames). Although the start of the prediction still affects the performance, the difference is not as large as it is for the correct predictions of rogue waves.

5.4 Summary

We studied behavior of rogue wave events and developed a model with ESN framework in predicting extreme events (rogue wave) in complex plasmas. One major question in

forecast of rogue waves is to predict whether rogue waves will occur or not, and if yes then when they will occur. In the present work, we first checked the second question by using testing sets that contains rogue wave events. We managed to achieve high successful rate in rogue wave prediction ($> 80\%$), when it starts at crest value. Yet, the accuracy of prediction declines significantly, if prediction starts at the mean intensities. Since the data is nearly symmetric about the mean intensity, the possibilities of two motions (up and down) are similar. Although we already introduced the lagged coordinates, which solve the problem to an extent, ESNs still might make false predictions as it goes into the wrong direction at the beginning.
In addition, the question about whether the events will happen is examined by testing on regular waves with no extreme events. To do that, the network makes a forecast for 1000 frames. If forecast contains rogue wave events within the first 50 frames, it is considered as a false prediction. On average, false prediction rate is around 20%, which is similar to the possibility of finding an extreme event with in 50 frames.
The next step would be to further break the symmetry of the data. For instance, by adding derivative of the data at each frame. Another way is to introduce a hybrid network that contains simple models which could simulate some behaviors in complex plasma, so that the predictions could be adjusted.

6 Discussion

In this thesis, we demonstrated the feasibility of forecast with Echo State Network (ESN) in simulation of real data with applications in Complex Plasmas. The tasks are divided into three parts, namely the prediction in spatial evolution, climate and short-term forecast of extreme (rogue wave) events.

To justify our choice of using ESN, we compared the performance of its long-term forecast with two other machine learning techniques, Artificial Neuron Network with forward feeding scheme (ANN) and Recurrent Network with Long-short-term Memory (LSTM). For this purpose, all the methods are trained and tested with simulated trajectories of the Lorenz system. The Correlation Dimension and Maximum Lyapunov Exponent of the predictions are computed to check whether the prediction captures the general statistic of the Lorenz system. We found that ESN and LSTM could reproduce the climate with stable performance, but about 30% predictions of ANN failed. In addition, the efficiencies of these techniques are measured by recording the time required for training and testing. The results showed that ESN requires the least of time to achieve similar forecast as the other two methods.

We showed that the spatial movement along the y-axis of a plasma wave could be forecasted. Since the waves propagate downward, intensities at the smallest y-values at each time frame is passed into the reservoir. Based on these values, the network predicts its spatial evolution. We exanimated the performance with different resolutions (intensity averaged over every 10 pixel and at single pixel on y-axis). In both cases, ESNs can predict the spatial movement with relatively similar result. The accuracy is slightly higher in the region where the movement is linear.

We have noticed that the wave disintegrates around $y = 2.7$ mm. To further investigate the behavior, we did cross validations with respect to different regions on y-axis. ESN is forced to predict the movement at the region where the wave disintegrates, after being trained with regular data. We found that the results obtained with different training and testing sets have large errors, but the general structure of the wave is captured. Therefore, we corrected the predicted mean intensity and the amplitude, by computing these values for training and testing sets. After the corrections, the performances with different training sets are almost same. The error when using training and testing sets from different regions are similar. It provides evidence that the underlying mechanism for the spatial movement of plasma wave is similar, even though the pattern disintegrates.

We tested the performance of long-term forecast. To smooth out high frequency fluctuations in the experimental data, rolling average is applied to the datasets. To quantify the changes, we calculated the number of waves with respect to its height (i.e., the difference between the intensity of the peak and the previous trough). We found that initially the height distribution is denser around 0, representing the existence of noise. After smoothing, the distribution shifts to the right and a second peak with larger height could be observed. ESNs with different spectral radius ρ are tested. The performance is evaluated by calculating the RMS error and correlation dimension of the predictions. From the results, we noticed a shift with the increment of ρ. With $\rho < 0.7$, the predictions always fail to reproduce the climate and decay into a constant function. When ρ is around 0.9, the prediction becomes very unstable. About 70% of the forecasted trajectories has much

larger amplitude than the testing data. However, some of the predictions ($\sim 10\%$) capture the long-term behavior. When ρ is further increased to 1.3, the predictions become stable again and produce correlation dimensions close to the experimental data. After optimization of the hyperparameters, we found that ESNs could reproduce the long-term statistics of plasma wave with $\rho > 1.3$. Although spectral radius below unity is often considered as a condition for an ESN to have the echo state property and it often works in practice [35], we found that, ESNs could reproduce the long-term statistics of plasma wave with $\rho > 1.3$.

We used similar setup of network and extended it to forecast extreme (rogue wave) events. First, we measured the statistics of the rogue wave events by computing the frequency of occurrence of these events and the significant height. Then, the hyperparameters of ESNs are adjusted such that the predictions have similar statistics as the experimental data. With the optimized ESN, we successfully forecasted more than 80% of the rogue wave events with a relatively low false predicting rate ($\sim 20\%$). This provides evidence that RCs, previously mainly used for numerical simulations, could also perform well in real-world experimental data, where the underlining physics is not so clear, and the measurements contain noise. We have also noticed that, if the prediction starts at crest values (peaks or troughs), the success ratio is much higher than, if it starts at zero-crossings (around mean intensities). This is caused by the symmetry in the plasma wave. Since the data is nearly symmetric about the mean intensity, the possibilities of two motions (up and down) are similar. Even though lagged coordinates are introduced, which solve the problem to an extent, RC still might make false predictions as it goes into the wrong direction at the beginning.

The next step would be to further break the symmetry of the data. For instance, by adding derivative of the data at each frame. Another way is to introduce a hybrid network that contains some simple models which could simulate the systems in complex plasma, so that the predictions could be adjusted.

Overall, our work demonstrates that the reservoir computing model is capable of forecast of nonlinear systems in Complex Plasmas. We found that much care should be placed on the choice of hyperparameters, as different choices of these parameters would have significant influence on the performance. Since we focused on a fully data-driven approach, our model could be easily applied to data in other complex dynamical systems. Thus, the research presented here could open new avenues for the climate prediction and the forecast of rogue waves and extreme events in complex systems.

References

[1] P. Bajaj. Spatial distribution of dust density wave properties in fluid complex plasmas (accepted). *PRE*, 2021.

[2] Andrew F. Bennett and Michael A. Thorburn. The generalized inverse of a nonlinear quasigeostrophic ocean circulation model. *Journal of Physical Oceanography*, 22(3):213 – 230, 1992. doi: 10.1175/1520-0485(1992)022⟨0213:TGIOAN⟩2.0.CO;2. URL https://journals.ametsoc.org/view/journals/phoc/22/3/1520-0485_1992_022_0213_tgioan_2_0_co_2.xml.

[3] A. Chattopadhyay, P. Hassanzadeh, and D. Subramanian. Data-driven predictions of a multiscale lorenz 96 chaotic system using machine-learning methods: reservoir computing, artificial neural network, and long short-term memory network. *Nonlinear Processes in Geophysics*, 27(3):373–389, 2020. doi: 10.5194/npg-27-373-2020. URL https://npg.copernicus.org/articles/27/373/2020/.

[4] P. Grassberger and I. Procaccia. Measuring the strangeness of strange attractors. *Physica D: Nonlinear Phenomena*, 9(1-2):189–208, 1983.

[5] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):855–868, 2009. doi: 10.1109/TPAMI.2008.137.

[6] Morton E. Gurtin and Richard C. MacCamy. Some simple models for nonlinear age-dependent population dynamics. *Mathematical Biosciences*, 43(3):199–211, 1979. ISSN 0025-5564. doi: https://doi.org/10.1016/0025-5564(79)90049-X. URL https://www.sciencedirect.com/science/article/pii/002555647990049X.

[7] J. Herteux and C. Räth. Breaking symmetries of the reservoir equations in echo state networks,. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(12):123142, 2020.

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

[9] Jie Huang, Wengang Zhou, Qilin Zhang, Houqiang Li, and Weiping Li. Video-based sign language recognition without temporal segmentation. *CoRR*, abs/1801.10111, 2018. URL http://arxiv.org/abs/1801.10111.

[10] Herbert Jaeger. The" echo state" approach to analysing and training recurrent neural networks-with an erratum note'. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148, 01 2001.

[11] Herbert Jaeger and Harald Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004. doi: 10.1126/science.1091277. URL https://www.science.org/doi/abs/10.1126/science.1091277.

[12] H. Kantz and T. Schreiber. *Nonlinear time series analysis / 2nd ed.* Cambridge university press, 2004.

[13] Po-Cheng Lin and Lin I. Synchronization of multiscale waveform focusing for rogue wave generation in dust acoustic wave turbulence. *Phys. Rev. Research*, 2(2), apr 2020. doi: 10.1103/physrevresearch.2.023090.

[14] Bin Liu, J. Goree, T. M. Flanagan, Abhijit Sen, Sanat Kumar Tiwari, Gurudas Ganguli, and Chris Crabtree. Experimental observation of cnoidal waveform of nonlinear dust acoustic waves. *Physics of Plasmas*, 25(11):113701, 2018. doi: 10.1063/1.5046402.

[15] Chaojun Liu, Yongqiang Wang, Kshitiz Kumar, and Yifan Gong. Investigations on speaker adaptation of lstm rnn models for speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5020–5024, 2016. doi: 10.1109/ICASSP.2016.7472633.

[16] Z. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott. Reservoir observers: Model-free inference of unmeasured variables in chaotic systems,. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(4):041102, 2017.

[17] Z. Lu, B. R. Hunt, and E. Ott. Attractor reconstruction by machine learning,. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(6):061104, 2018. doi: 10.1063/1.5039508. URL https://doi.org/10.1063/1.5039508.

[18] W. Maass, T. Natschlaeger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations,. *Neural Computation*, 14(11):2531–2560, 2002. doi: 10.1162/089976602760407955.

[19] André Melzer. *Physics of Dusty Plasmas*, volume 962. Springer, 2019.

[20] Robert L. Merlino. 25 years of dust acoustic waves. *J. Plasma Phys.*, 80:773, 2014. doi: 10.1017/S0022377814000312.

[21] Gregor E. Morfill and Alexei V. Ivlev. Complex plasmas: An interdisciplinary research field. *Rev. Mod. Phys.*, 81:1353, 2009. doi: 10.1103/RevModPhys.81.1353.

[22] W. M. Moslem, R. Sabry, S. K. El-Labany, and P. K. Shukla. Dust-acoustic rogue waves in a nonextensive plasma. *Phys. Rev. E*, 84(6):066402, 2011. doi: 10.1103/physreve.84.066402.

[23] P. Müller, C. Garrett, and A. Osborne. Rogue waves. *Oceanography*, 18, September 2005. URL https://doi.org/10.5670/oceanog.2005.30. 66.

[24] Ali Bou Nassif, Ismail Shahin, Imtinan Attili, Mohammad Azzeh, and Khaled Shaalan. Speech recognition using deep neural networks: A systematic review. *IEEE Access*, 7:19143–19165, 2019. doi: 10.1109/ACCESS.2019.2896880.

[25] Sandeep Pandey and Jörg Schumacher. Reservoir computing model of two-dimensional turbulent convection. *Phys. Rev. Fluids*, 5:113506, Nov 2020. doi: 10.1103/PhysRevFluids.5.113506. URL https://link.aps.org/doi/10.1103/PhysRevFluids.5.113506.

[26] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Phys. Rev. Lett.*, 120:024102, Jan 2018. doi: 10.1103/PhysRevLett.120.024102. URL https://link.aps.org/doi/10.1103/PhysRevLett.120.024102.

[27] Zhilin Qu, Gang Hu, Alan Garfinkel, and James N. Weiss. Nonlinear and stochastic dynamics in the heart. *Physics Reports*, 543(2):61–162, 2014. ISSN 0370-1573. doi: https://doi.org/10.1016/j.physrep.2014.05.002. URL https://www.sciencedirect.com/science/article/pii/S037015731400204X. Nonlinear and Stochastic Dynamics in the Heart.

[28] M. T. Rosenstein, J. J. Collins, and C. J. De Luca. A practical method for calculating largest lyapunov exponents from small data sets. *Physica D*, 65(1-2):117–134, 1993.

[29] Mierk Schwabe, Sergey K Zhdanov, Hubertus M Thomas, Alexei V Ivlev, Milenko Rubin-Zuzic, Gregor E Morfill, Vladimir I Molotkov, Andrey M Lipaev, Vladimir E Fortov, and Thomas Reiter. Nonlinear waves externally excited in a complex plasma under microgravity conditions. *New J. Phys.*, 10:033037, 2008. doi: 10.1088/1367-2630/10/3/033037.

[30] D. Solli, C. Ropers, P. Koonath, and B. Jalali. Optical rogue waves. *Nature*, 450, September 2007. doi: https://doi.org/10.1038/nature06402.

[31] Peter Tino and Georg Dorffner. Predicting the future of discrete sequences from fractal representations of the past. *Machine Learning*, 45(2):187–217, 2001.

[32] Ya-Yi Tsai, Jun-Yi Tsai, and Lin I. Generation of acoustic rogue waves in dusty plasmas through three-dimensional particle focusing by distorted waveforms. *Nature Phys.*, 12:573–577, 2016. doi: 10.1038/nphys3669.

[33] D. Vassiliadis, A.J. Klimas, J.A. Valdivia, and D.N. Baker. The nonlinear dynamics of space weather. *Advances in Space Research*, 26(1):197–207, 2000. ISSN 0273-1177. doi: https://doi.org/10.1016/S0273-1177(99)01050-9. URL https://www.sciencedirect.com/science/article/pii/S0273117799010509. Space Weather: Physics and Applications.

[34] P. Vlachas, W. Byeon, Z. Wan, T. Sapsis, and Koumoutsakos P. Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *P. Roy. Soc. A-Math*, 474(20170844), 2018. doi: https://doi.org/10.1098/rspa.2017.0844.

[35] Izzet Yildiz, Herbert Jaeger, and Stefan Kiebel. Re-visiting the echo state property. *Neural networks : the official journal of the International Neural Network Society*, 35:1–9, 07 2012. doi: 10.1016/j.neunet.2012.07.005.

[36] G.Peter Zhang, B.Eddy Patuwo, and Michael Y. Hu. A simulation study of artificial neural networks for nonlinear time-series forecasting. *Computers & Operations Research*, 28(4):381–396, 2001. ISSN 0305-0548. doi: https://doi.org/10.1016/S0305-0548(99)00123-9. URL https://www.sciencedirect.com/science/article/pii/S0305054899001239.

[37] Yudong Zhang, Shuihua Wang, Genlin Ji, and Preetha Phillips. Fruit classification using computer vision and feedforward neural network. *Journal of Food Engineering*, 143:167–177, 2014. ISSN 0260-8774. doi: https://doi.org/10.1016/j.jfoodeng.2014.07.001. URL https://www.sciencedirect.com/science/article/pii/S026087741400291X.

[38] Li-Chen Zhao. Dynamics of nonautonomous rogue waves in bose–einstein condensate. *Annals of Physics*, 329:73–79, 2013. ISSN 0003-4916. doi: https://doi.org/10.1016/j.aop.2012.10.010. URL https://www.sciencedirect.com/science/article/pii/S0003491612001716.

Declaration of authorship

I hereby declare that the report submitted is my own unaided work. All direct or indirect sources used are acknowledged as references. I am aware that the Thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the report as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future Theses submitted. Further rights of reproduction and usage, however, are not granted here. This paper was not previously presented to another examination board and has not been published.

Location, date

Name