2021 IEEE Symposium Series on Computational Intelligence (SSCI) | 978-1-7281-9048-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/SSCI50451.2021.9659913

Second-order Time Delay Reservoir Computing for Nonlinear Time Series Problems

Xinming Shi[1, 2], Jiashi Gao[1], Leandro L. Minku[2], James J. Q. Yu[1] and Xin Yao[1, 2]
[1]*Department of Computer Science and Engineering, Southern University of Science and Technology*
[2]*School of Computer Science, University of Birmingham*
Shenzhen, China, and Birmingham, UK
xxs9722@cs.bham.ac.uk

***Abstract*—Time Delay Reservoir (TDR) can exhibit effects of high dimensionality and short-term memory based on delay differential equations (DDEs), as well as having hardware-friendly characteristics. However, the predictive performance and memory capacity of the standard TDRs are still limited, and dependent on the hyperparameter of the oscillation function. In this paper, we first analyze these limitations and their corresponding reasons. We find that the reasons for such limitations are fused by two aspects, which are the trade-off between the strength of self-feedback and neighboring-feedback caused by neuron separation, as well as the unsuitable order setting of the nonlinear function in DDE. Therefore, we propose a new form of TDR with second-order time delay to overcome such limitations, incurring a more flexible time-multiplexing. Moreover, a parameter-free nonlinear function is introduced to substitute the classic Mackey-Glass oscillator, which alleviates the problem of parameter dependency. Our experiments show that the proposed approach achieves better predictive performance and memory capacity compared with the standard TDR. Our proposed model also outperforms six other existing approaches on both time series prediction and recognition tasks.**

I. INTRODUCTION

Reservoir computing (RC) is a computing paradigm in the field of machine learning [1]. RC has different variants since it was firstly invented, such as Echo State Network (ESN) [2] and Liquid State Machine (LSM) [3]. It has been widely applied in the time series problems [4] [5] [6] [7] [8]. Both of these two variants are originally composed of fixed randomly-generated reservoir architectures and neuron weights, where a reservoir with H neurons will have up to H^2 connections, potentially incurring a large area and power overhead [9]. Moreover, the paradigms of RC with randomly-generated topology are also complex to implement in hardware, where the routing complexity, area, and power consumption are high [10]. Another category of RC called Time Delay Reservoirs (TDR) can prevent the large overhead of traditional ESN by time multiplexing resources [9], while satisfying the following desirable properties of RC [11], [12]: (1) RCs should nonlinearly transform the input signal into a high-dimensional state space. (2) The dynamics of the reservoir should be such that it exhibits a short-term memory. (3) The results of RC computations must be reproducible and robust against noise.

Due to the time multiplexing of neuron states, TDRs incur less neuron connections compared with random-connection reservoir and have friendly features to hardware implementation [13]. Generally, a simple first-order delay dynamic system could be applied to model the TDR, tackling different tasks. For example, L. Appeltant *et al.* [12] proposed a special TDR utilizing a single neuron and a delayed feedback to fulfil the demands required of reservoirs. L. Grigoryeva *et al.* [13] solved the stochastic nonlinear time series forecasting problem by constructing a time-delay dynamic system with a first-order inertial element and a delay element. L. Keuninckx *et al.* [14] tackled a real-time audio processing task by applying a cascade of TDR with a discrete form of first-order delay system. Since the first-order delay dynamic system could be modelled by Delay Differential Equations (DDEs), TDR is amenable to a large number of experimental hardware implementations [11], such as FPGA (Field Programmable Gate Array) TDR [15], optoelectronic TDR [16] and memristive TDR [17], while the performance of TDR with first-order delay dynamic system may potentially be not enough.

The parameters of TDR play a significant role on its performance. Researchers [12] have clarified the analogue relationship between the parameters of random-connected ESNs and TDRs modelled by dynamic system, where the input gain, feedback gain, and delay time in TDR are related to input scaling, spectral radius, and sparsity of the adjacent matrix in ESN, respectively. There have been related work on selecting the optimal parameter values for TDR [12], [18], however, how to further improve the predictive performance and memory capacity of TDR under the existing optimal parameter values and how to prevent the parameter dependence of the predictive performance of TDR are still needed to be taken into consideration. Therefore, we propose a novel TDR to solve these problems from two aspects. First, a second-order time delay approach is designed to model TDR making the time multiplexing more flexible, which promotes the memory capacity and further improves predictive performance. Second, a parameter-free nonlinear function is applied to the proposed second-order time delay system, reducing the number of parameters to be tuned in the TDR. Our experiments show that the proposed approach obtains good results both in time series prediction and recognition tasks. In summary, the contributions of this work are summarised as follows:

- We provide an analysis of the problems of standard TDR,

978-1-7281-9048-8/21/$31.00 ©2021 IEEE

showing that the predictive performance and memory capacity are still limited and parameter-dependent. The causes of these problems are also given.

- We propose a second-order time delay reservoir for the time series problem, leading to a more flexible time-multiplexing, which improves the predictive performance and memory capacity of TDR.
- Combined with this second-order time delay, we introduce a parameter-free oscillator with exponential decay to substitute the classic Mackey-Glass oscillator, avoiding the need to tune the nonlinearity degree in the proposed TDR.
- Through experiments with artificial and real-world datasets, we show the second-order time delay reservoir computing outperforms standard TDR and some existing approaches on both time series prediction and recognition tasks in terms of predictive performance and memory capacity.

The rest of the paper is organized as follows: Section II introduces the formulation of the standard TDR [12], and analyzes its existing problems as well as their corresponding reasons. Section III proposes the second-order time delay reservoir computing (STDR). Section IV evaluates and discusses the proposed STDR from different aspects via artificial and real-world data sets. At the end, Section V concludes the paper.

II. Problem Analysis of Standard TDR

In this section, we first gives the formulation of standard TDR in Section II-A. And then, the main problems of standard TDR will be analyzed and their corresponding reasons will be given in Section II-B and Section II-C. The main problems are limited memory capacity and predictive performance even when the optimal hyperparameters are used.

A. Formulation of Standard TDR

The states of the reservoir in TDR can be described generally by the solutions of the following DDE:

$$\dot{x}(t) = -x(t) + f(x(t-\tau), I(t)), \tag{1}$$

where $x(t)$ refers to the states of reservoir, $I(t)$ is the input signal connected to the reservoir, and f refers to a nonlinear function. With delay interval τ, N equidistant points will be separated in time by $\theta = \tau/N$, and these N equidistant points could be regarded as *virtual neurons* being multiplexed in the given time scale. By Euler discretization of Equation (1) with integration step θ, the reservoir state $x_i(k)$ could be rewritten as:

$$x_i(k) = \frac{1}{1+\theta} x_{i-1}(k) + \frac{\theta}{1+\theta} f(x_i(k-1), I_i(k)). \tag{2}$$

The reservoir state $x_i(k)$ is not only related to the state of previous neuron $x_{i-1}(k)$ but also the state $x_i(k-1)$ of the neuron in previous layer and triggered by the corresponding reservoir input as depicted in Figure 1.

Based on Equation (2), we can see that there are two parameters that can influence the performance of standard TDR – the neuron separation θ and the nonlinear function f. The neuron separation θ controls the strength of neuron self-feedback and neighboring-feedback and the nonlinear function f controls the type and degree of nonlinearity. In the next subsections, we analyse how these two factors lead to such limitations. The specific setup of the experiments involved this analysis will be introduced in Section IV-B, as it becomes more relevant in that section.

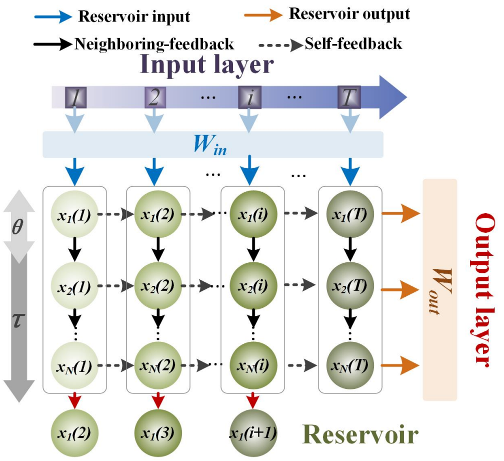


Fig. 1. Basic structure of standard TDR.

B. Hyperparameter Selection of θ

Fig. 2(a) shows the normalized mean squared error (NMSE) [11] for a 10th order nonlinear auto-regressive moving average (NARMA) [19] task as a function of node separation θ, where we can find that smaller or larger θ both lead to performance degradation, and when $\theta = 0.2$, TDR can perform well. Therefore, $\theta = 0.2$ has been regarded as the optimal hyperparameter value for TDR [12]. However, with $\theta = 0.2$, the memory capacity of TDR (depicted in Fig. 2(b)) is limited to within 10^1 lags. This means that TDR is only able to have good memory capacity (10^0) for recalling the previous 10^1 steps history, being suboptimal for tasks that require longer memory. The specific definition of memory capacity will be given in Section IV-B1.

We further analyze the reasons of the above mentioned problems, which are concluded as follows:

- Smaller θ will weaken the self-feedback between neurons. As Equation (2) indicates, when θ is smaller, the latter term of Equation (2) tends to be zero, causing the neuron state to loose the dependency on the delay and on the input to the neuron state.
- Larger θ will lead to invalid Euler discretization. When θ is larger, the first-order difference quotient cannot replace the first-order derivation in Equation 1. Moreover, as Equation (2) indicates, a larger θ may incur too much self-feedback and less neighboring-feedback such that the system behaves like independent nodes, each of which is coupled only to itself at the previous time step.

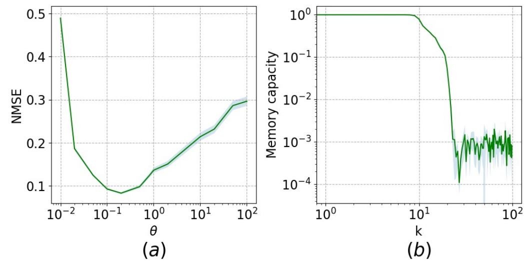


Fig. 2. (a) NMSE of standard TDR with varying neuron separation θ in Santa Fe Laser prediction task. (b) Memory capacity (from Equation (18)) corresponding to varying lags k when using $\theta = 0.2$. The shadowed areas of each curve indicate standard deviation (20 runs).

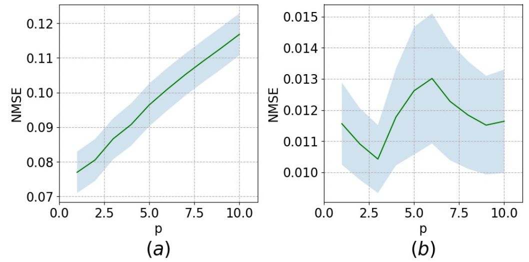


Fig. 3. Performance of standard TDR with varying p for (a) 10th order NARMA task; (b) Santa Fe Laser prediction task. The shadowed areas of each curve indicate standard deviation (20 runs).

- The optimal value of θ still cannot satisfy the requirement of short-term memory capacity for the tasks with high-order dynamic systems, as discussed in the beginning of this section and illustrated in Fig. 2.

Based on the problems and their reasons, it is valuable to further improve the memory capacity of TDR.

C. Selection of Nonlinear Type

In TDR, f applied in Equation (2) is the Mackey-Glass oscillator [20]:

$$f(x) = \frac{\eta x}{(1+x)^p}, \tag{3}$$

where η is the feedback gain and the exponent p refers to the nonlinear type of TDR. As explained by Appeltant *et al.* [12], the exponent p plays an important role in changing the nonlinearity of TDR, where smaller p indicates weaker nonlinearity and longer memory, while larger p leads to higher degree of nonlinearity. Fig. 3 shows how the NMSE of the standard TDR varies with the exponent p of the Mackey-Glass oscillator for two time series prediction tasks.

As for 10th order NARMA task shown in Fig. 3(a), when $p = 1$, the standard TDR achieved its best NMSE. However, larger values of p led to a dramatic increase in NMSE. This is because the predictive performance for the 10th order NARMA task relies on longer memory instead of higher nonlinearity. So, when $p = 1$ (which indicates longer memory), the TDR can perform best. As we can see, the optimal value of p may be different depending on the task being solved and its underlying characteristics. As for the Santa Fe Laser prediction task shown in Fig. 3(b), we also can see the NMSE differences between $p = 3$ and $p = 6$.

Based on the analysis above, the value of the hyperparameter p can have a significant impact on predictive performance, which is highly dependent on the task being solved. It is valuable to research how to achieve good predictive performance without having to rely on hyperparameters such as p.

III. Second-order Time Delay Reservoir Computing

In this section, we introduce a second-order TDR to tackle the mentioned-above problems, where Section III-A gives the structure of second-order TDR, Section III-B introduces the input layer of proposed model, and Section III-C gives the construction of readout layer, respectively.

A. Construction of Second-order Time Delay Reservoir

In order to solve the limited predictive performance and memory capacity in standard TDR, we propose a time-delay reservoir with multi-time multiplexing by introducing another delay τ_{jump}, which can establish the connection between the current neuron and another neuron at different locations in the previous layer(s). The approach is equipped with more flexible time multiplexing and allows for the representation of various dynamical timescales, specifically, the current neuron state not only relates to the two states mentioned in Equation (2) ($x_{i-1}(k)$ and $x_i(k-1)$), but also to states from different locations in the previous layer, corresponding to a delay of τ_{jump}. Therefore, the delay differential equation in our method is described as follows:

$$\dot{x}(t) = -x(t) + f(x(t-\tau), x(t-\tau-\tau_{jump}), I(t)). \tag{4}$$

The two types of time delay are represented as follows:

$$\tau = N\theta, \quad (5) \qquad \tau_{jump} = J\theta, \quad (6)$$

where N is the number of nodes in the reservoir and θ is the separation between neurons. Given that jump step J is an integer within $(0, N)$, Equation (4) can be rewritten by Euler discretization with integration step $\theta := \tau/N$ as follows:

$$x_i = \begin{cases} \frac{x_{i-1}(k)}{1+\theta} + \frac{\theta f(x_i(k-1), x_{i-J}(k-1), I_i(k))}{1+\theta} & i \geq J+1 \\ \frac{x_{i-1}(k)}{1+\theta} + \frac{\theta f(x_i(k-1), x_{N-(J-i)}(k-2), I_i(k))}{1+\theta} & i < J+1. \end{cases} \tag{7}$$

In the equation above, $x_i(k)$ is the i-th neuron value of the k-th layer of the reservoir, $x_i(k-1)$ refers to the self-feedback, which is the state of the neuron in the same position of $k-1$ layer, and $x_{i-1}(k)$ denotes the neighboring-feedback, which is the state of the neuron in the previous $(i-1)$ position in the same layer. This is depicted in Fig. 4, which illustrates the proposed approach. For different values of $i = 1, 2, ..., N$, three different scenarios are depicted in Fig. 4, where the first two layers describe the scenario of $i < J+1$, the middle two layers describe the scenario of $i = J+1$, and the last two layers describe the scenario of $i > J+1$.

As shown in Fig. 4, within an interval τ, the DDE-based TDR is discretized as N *virtual neurons* in the vertical direction, and these *virtual neurons* are all history-dependent, so that history states could be transferred in the horizontal direction. In addition, the neuron states in the previous one or two layers could also be transferred to the current state.

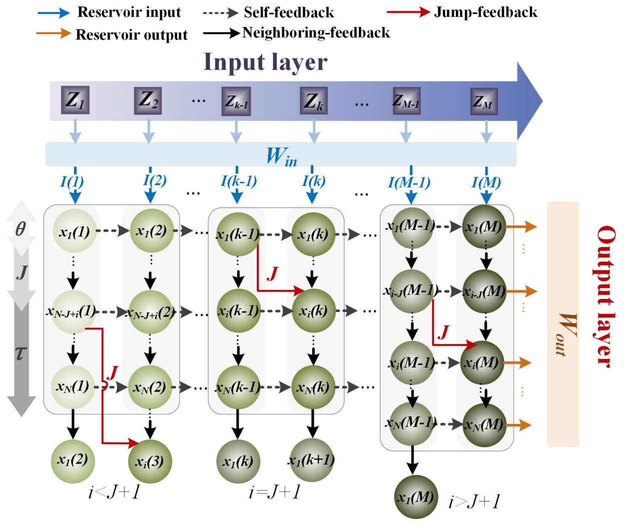

Reservoir input
Self-feedback
Jump-feedback
Reservoir output
Neighboring-feedback
Input layer
Z_1 Z_2 ... Z_{k-1} Z_k ... Z_{M-1} Z_M
W_{in}
I(1) I(2) ... I(k-1) I(k) ... I(M-1) I(M)
θ
J
τ
$x_1(1)$ $x_1(2)$ $x_1(k-1)$ $x_1(k)$ $x_1(M-1)$ $x_1(M)$
$x_{N-J+i}(1)$ $x_{N-J+i}(2)$ $x_i(k-1)$ $x_i(k)$ $x_{i-J}(M-1)$ $x_{i-J}(M)$
$x_i(M-1)$ $x_i(M)$
$x_N(1)$ $x_N(2)$ $x_N(k-1)$ $x_N(k)$ $x_N(M-1)$ $x_N(M)$
$x_1(2)$ $x_i(3)$ $x_1(k)$ $x_1(k+1)$ $x_1(M)$
J
$i<J+1$
$i=J+1$
$i>J+1$
W_{out}
Output layer

Fig. 4. Diagram of second-order time delay reservoir computing.

Therefore, the current state $x_i(k)$ in the second-order time delay reservoir is traced form three components:

- Neighboring-feedback $x_{i-1}(k)$ in the same layer, denoted by black solid vertical arrows in Fig. 4;
- Self-feedback $x_i(k-1)$ in the previous layer, denoted by black dotted horizontal arrows in Fig. 4;
- Jump-feedback $x_{i-J}(k-1)$ or $x_{N-(J-i)}(k-2)$ with the interval J, denoted by red solid arrows in Fig. 4.

For the sake of decoupling the exponent p of the Mackey-Glass oscillator from the nonlinearity of the reservoir, we introduce a parameter-free oscillation function with exponential decay as the nonlinear function f in Equation (4), which is described as:

$$f(x) = e^{-x} \sin x. \tag{8}$$

Different from the Mackey-Glass oscillator (Equation (3)) that has been generally used as f [12], this new nonlinear function can be expanded to infinite order using Taylor expansions, $\sum_{n=0}^{\infty} \frac{(-x)^n}{n!} \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!}$, which is an analytic function that convergences to the derivatives of arbitrary orders. Therefore, there is no requirement of tuning the hyperparameter p of the Mackey-Glass oscillator to obtain different nonlinear fitting for different tasks. In addition, the exponential decay of the newly-applied oscillation function also matches the fading memory of reservoir's features.

B. Construction of Input Layer

As shown in Fig. 4, the time series sample $z = z_1, ..., z_M$ is given as input. Before the sample is imported into the reservoir, there is a random input mask W_{in} that will produce a new signal I, which can be described as:

$$I = W_{in} z, \tag{9}$$

where $W_{in} \in \mathbb{R}^{M \times N}$. This process has two effects:

- The input mask distributes the information contained in the same time series value into all neurons and it makes the dimensional multiplexing of the input.
- The mask values with zero mean make the input time series z with non-zero mean to be zero; such property is convenient for eliminating the intercept in the ridge regression.

C. Construction of Readout Layer

With the input signal of reservoir $I(t)$, there is a corresponding teaching signal $y(t) \in \mathbb{R}^M$ and a M-dimensional output could be obtained by output matrix and reservoir state $\hat{y}(t) := x(t)^{\mathsf{T}} \times W_{out}$. The training process will find the output matrix $W_{out} \in \mathbb{R}^{N \times M}$ by minimizing the distance between the output and the teaching signal, which is described as the following optimization problem:

$$W_{out} := \arg\min_{W} \left(\sum_{i=1}^{M} ||x(i)^{\mathsf{T}} \times W - y(i)||^2 + \lambda ||W||^2 \right) \tag{10}$$

where $||W||^2$ refers to a regularisation term to prevent overfitting, and λ controls its intensity. In order to optimize this problem, ridge regression has been applied, whose solution could be given by:

$$W_{out} = (XX^{\mathsf{T}} + \lambda I)^{-1} X y. \tag{11}$$

IV. Experiments

In this section, we first conduct experiments to compare the predictive performance resulting from the better memory capacity and parameter-free nonlinear function of the proposed STDR against the standard TDR. We ran both STDR and TDR with Mackey-Glass (MG) and with our proposed exponential nonlinear function (EXP) to support the analysis. Then, we conduct comparisons with several benchmark models for time series prediction, namely vanilla ESN [2], vanilla LSTM [21], vanilla RNN [22], memory-augmented LSMT and memory-augmented RNN [23]. As for the later two, another parameter D is introduced for the memory-augment, therefore, memory-augmented LSMT and memory-augmented RNN will be abbreviated as mLSTM_FIXD and mRNN_FIXD in the following Sections. The hyperparameters of these approaches are set as the same way with their corresponding literatures.

A. Time Series Tasks

This section explains the 155 time series tasks used in our experiments, including 2 system identification tasks, 2 time series prediction tasks, 1 spoken digit recognition task, and 150 memory mapping tasks.

1) System Identification Task: In the system identification task, we considered the NARMA systems of order 10 and 20 respectively [19]:

$$\begin{aligned} y(t+1) = & 0.3y(t) + 0.05y(t) \sum_{i=0}^{9} y(t-i) \\ & + 1.5s(t-9)s(t) + 0.1, \end{aligned} \tag{12}$$

$$y\left(t+1\right) = \tanh\left(0.3y\left(t\right) + 0.05y\left(t\right)\sum_{i=0}^{19} y\left(t-i\right) \right. \qquad (13)$$
$$\left. + 1.5s\left(t-19\right)s\left(t\right) + 0.1\right),$$

where $s\left(t\right)$ is the random input series ranged from $\left[0, 0.5\right]$ and $y\left(t\right)$ is the output of the system. NARMA tasks aim at measuring the ability of a neural network to model nonlinear and long-term memory systems. We selected the NARMA sequence with 8000 items, where the first 2000 were used as the training set, the following 4000 were validation set, and the remaining were testing set, the first 200 items of them were used as the washout.

2) Time Series Prediction Tasks on Santa Fe Laser data set: For the task of time series prediction, the Santa Fe Laser data set [24] was used [1], which is a cross-cut through periodic to chaotic intensity pulsations of a real laser. This task is to predict the next value of the input sequence. Two different Santa Fe datasets were used, the first of which is the univariate time series A derived from laser-generated data, and the second is the computer-generated time series D. For both time series A and D, we selected the training, validation, testing sets and washout as in the system identification tasks.

3) Time Series Prediction Tasks on Nonlinear Audio prediction: We also applied a nonlinear audio prediction task, which is to predict the future samples under the given history horizon. We selected the training, validation, testing sets and washout as in the system identification tasks

4) Memory and Nonlinear Mapping Tasks: This task [25] is used to study two characteristics of the reservoir: memory and the capacity of processing nonlinearities in the input time series. The input signal $s(t)$ is an uncorrelated uniform distribution over the range $[-0.8, 0.8]$. The task is to reconstruct the delayed and nonlinear system as follows under the input of $s(t)$:

$$y_{p,d}(t) = sign[\beta(t-d)] \cdot \left|\beta(t-d)\right|^{p}, \qquad (14)$$

where the delay ($d > 0$) controls the memory and the index term ($p > 0$) controls the nonlinearity of the system. $\beta(t-d)$ is the product of two delayed successive inputs:

$$\beta(t-d) = s(t-d) \cdot s(t-d-1). \qquad (15)$$

We use 150 different readouts corresponding to different combinations of $d = 1, ..., 15$ and $p = 1, ..., 10$. The training, validation, testing sets and washout were selected as in the system identification tasks.

5) Classification Task on Spoken Digit: In this task, we applied an isolated spoken digit recognition data set obtained from $kaggle$ [2], which contains 1500 spoken isolated digits from 0 to 9, and each digit is spoken 50 times by three male speakers. Due to its limited number of items, 1350 items are used for training and 150 for testing, and 10-fold cross validation was performed. The Mel-Frequency Cepstral Coefficients (MFCC) is used for feature extraction in wave signals.

6) Classification Task on Chlorine Concentration Dataset: It models the hydraulic and water quality behavior of water distribution piping systems. The data set consists of 166 nodes (pipe junctions) and measurement of the Chlorine concentration level at all these nodes during 15 days, which obtained from UCR [3]. We set 300 items for training, 600 items for validation and 600 items for testing.

B. Experimental Setup

1) Quality Measures: Except for spoken digit recognition, which is a classification task, we adopt the normalized mean squared error (NMSE) as a measure of predictive performance in our experiments [11]:

$$NMSE = \frac{\left\langle \left\| \hat{y}(t) - y(t) \right\|^{2} \right\rangle}{\left\langle \left\| y(t) - \langle y(t) \rangle \right\|^{2} \right\rangle}, \qquad (16)$$

where $y(t)$ is the desired output (target), $\hat{y}(t)$ is the readout output, $\|.\|$ denotes the Euclidean norm, and $\langle . \rangle$ denotes the empirical mean. For spoken digit recognition, we use accuracy (the fraction of correctly classified samples).

Short-term memory capacity is used to quantify the ability of recurrent network architectures to encode past events in their state space so that past items can be recovered. Given an input signal $s(t)$, for a delay k, we used the trained network to conduct the task of outputting $s(t-k)$ after observing $s(t-k+1), ..., s(t-1), s(t)$. The degree of the fitting is then measured by:

$$MC_{k} = \frac{Cov^{2}(s(t-k), y(t))}{Var(s(t))Var(y(t))}, \qquad (17)$$

where $y(t)$ is the observed network output, Cov denotes the covariance and Var denotes variance. The short-term memory (STM) capacity is then given by [26]:

$$MC = \sum_{k=1}^{k_{max}} MC_{k}. \qquad (18)$$

Due to the property of short-term memory of reservoirs, the upper limit of the sum is set to $k_{max} = 100$ [27].

2) Reservoir Hyperparameter Selection: In the experiment, the value of W_{in} is randomly selected in $[-0.1, 0.1]$. By applying Equation (8) and making hyperparameters α, β and γ explicit as in [12], Equation (4) can be rewritten as:

$$\dot{x}(t) = -x(t) + \frac{\sin(\alpha x(t-\tau) + \beta x(t-\tau-\tau_{jump}) + \gamma I(t))}{e^{(\alpha x(t-\tau) + \beta x(t-\tau-\tau_{jump}) + \gamma I(t))}}, \qquad (19)$$

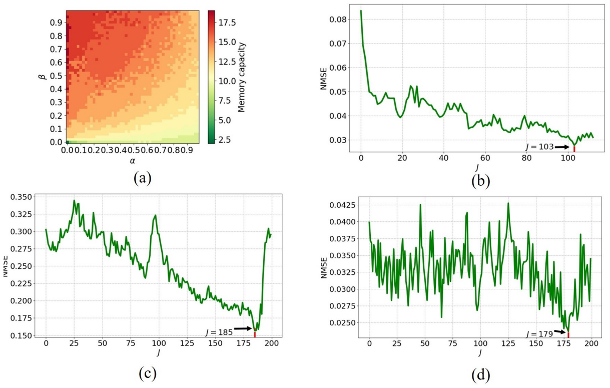
where α and β are hyperparameters indicating the trade-off between the strength of self-feedback and jump-feedback, and γ is a hyperparameter denoting the input gain. As in [12], the input gain γ is set as 0.05 for time series prediction tasks and 0.5 for recognition tasks. In addition, we also investigate the predictive performance and memory capacity of STDR with different α and β values on each task in order to tune these hyperparameter values.

[1] http://web.cecs.pdx.edu/ mcnames/DataSets/index.html

[2] https://www.kaggle.com/divyanshu99/spoken-digit-dataset

[3] www.cs.ucr.edu/∼eamonn/time_series_data

We first checked which values of α and β led to better NMSE values based on a grid search while fixing $J = 103$. We then checked the MC values corresponding to the different NMSE values (Fig. 5(a)) and chose the α and β values that best matched the desired nonlinear order for the task in hand whilst achieving good NMSE values. The value of $J = 103$ was chosen for leading to the best NMSE when using $\alpha = \beta = 0.6$ on 10th order NARMA (Fig. 5(b)), and $J = 185, 179$ for the 20th NARMA and Santa Fe Laser tasks, respectively. Based on this procedure, the following values were adopted: $\alpha = 0.6$ and $\beta = 0.6$ for the 10th order NARMA, $\alpha = 0.1$ and $\beta = 0.9$ for 20th order NARMA, $\alpha = 0.9$ and $\beta = 0.02$ for the Santa Fe Laser tasks, and $\alpha = 0.6$ and $\beta = 0.6$ for the memory and nonlinear mapping tasks. For each task, TDR's hyperparameters were tuned based on the procedure recommended in the literature [12].



(a) (b) (c) (d)

Fig. 5. (a) Memory capacity under different α and β; (b) NMSE under different J for 10-th order Narma task; (c) NMSE under different J for 20-th order Narma task; (d) NMSE under different J for Santa Fe.

We will present the average results across 20 runs for each approach on each task using three different reservoir sizes (N = 100, 200, and 300), except for the memory and nonlinear mapping tasks, where N = 100 due to the large number of experiments.

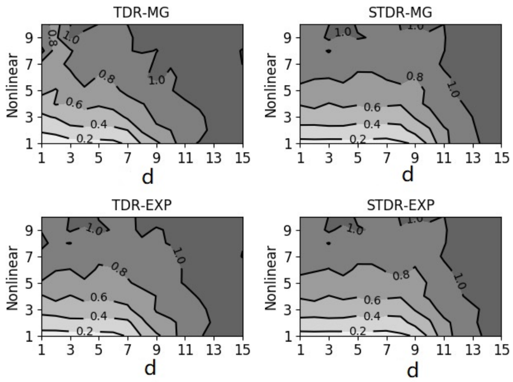
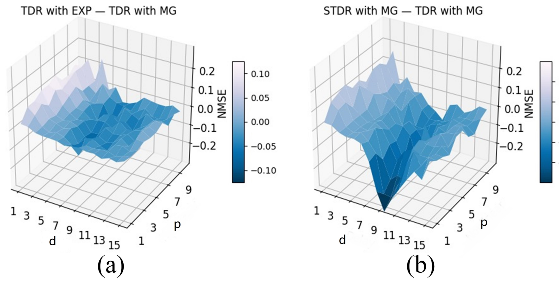


Fig. 6. NMSE obtained by TDR-MG, STDR-MG, TDR-EXP, and STDR-EXP for the memory and nonlinear mapping tasks with different delay d and nonlinear degree p.



(a) (b)

Fig. 7. Differences between NMSE values of (a) TDR-EXP and TDR-MG, (b) STDR-MG and TDR-MG for nonlinear mapping tasks with different delay d and nonlinear degree p.

C. Experimental Results

In this section, we first give the validation of the benefits of the proposed method compared with the standard TDR in Section IV-C1. Furthermore, the comparison results with other existing methods are also introduced Section IV-C2.

1) Validating the benefits of the proposed method over standard TDR: In order to verify the benefits of our proposed method over the standard TDR, the experiments of standard TDR and STDR with classic nonlinear function (MG) and our proposed one (EXP) are carried on, respectively. The tasks mentioned in Section IV-A are all applied for the validation. The average predictive performance for each task except the memory and nonlinear mapping tasks is presented in Table I. For the memory and nonlinear mapping tasks, the results are presented in Fig. 6 and 7, as there are 150 tasks.

Table I indicates that STDR outperforms the standard TDR. The results based on the proposed EXP nonlinear function appear slightly better than those based on MG.

By examining Fig. 6 to compare the results of TDR and STDR with the same nonlinear function, we can see that STDR has larger area corresponding to better NMSE (NMSE< 1.0), indicating that STDR has greater ability to fit wider ranges of delay and nonlinearity. Fig. 7(b) further shows that STDR also tends to generate better NMSE with increasing delay d compared with TDR. Fig. 7(a) also shows that the differences in NMSE obtained by TDR with EXP and MG are not so large as the differences obtained between STDR and TDR in Fig. 7(b).

In addition, we performed Mann–Whitney U tests with level of significance 0.05 to compare the predictive performance obtained by each methods across tasks, of which the results are shown in the Table II. Based on the comparisons TDR-EXP vs TDR-MG (p-value=0.04448), we can see that EXP provides some help to improve predictive performance. Based on the comparisons STDR-EXP vs TDR-EXP (p-value = 0.0002198) and STDR-MG vs TDR-MG (p-value=0.0001230), we can confirm that the proposed features of the STDR approach designed to obtain better memory capacity are helpful to improve predictive performance across tasks.

Therefore, the main feature of our proposed approach that

TABLE I

The results of TDR and STDR with different nonlinear function for different tasks. The best result is highlighted in bold.

Node size	Methods	10-th order Narma	20-th order Narma	Santa Fe Set-A	Santa Fe Set-D	Nonlinear Audio	Chlorine Concentration	Spoken Digit Recognition
		NMSE					Accuracy	
100	TDR-MG	0.1239	0.6245	0.0158	0.0231	0.0461	0.8700	0.9033
	TDR-EXP	0.1231	0.6245	0.0200	0.0232	0.0416	0.9025	0.9080
	STDR-MG	0.0509	0.3908	0.0173	0.0299	0.0310	0.8956	0.9106
	STDR-EXP	**0.0496**	**0.3897**	**0.0136**	**0.0229**	**0.03495**	**0.9280**	**0.9220**
200	TDR-MG	0.0788	0.3480	0.0115	0.0171	0.0607	0.8979	0.9046
	TDR-EXP	0.0774	0.3499	0.0124	0.0172	0.0476	0.9025	0.9106
	STDR-MG	0.0145	0.1842	0.0139	0.0171	0.0185	**0.9257**	0.9126
	STDR-EXP	**0.0143**	**0.1804**	**0.0110**	**0.0170**	**0.0156**	0.9164	**0.9233**
300	TDR-MG	0.0674	0.3470	0.0119	0.0141	0.1483	0.9095	0.9073
	TDR-EXP	0.0675	0.3441	0.0135	0.0151	0.3490	0.9025	0.9066
	STDR-MG	0.0118	0.1600	0.0116	0.0136	0.0267	0.9234	0.9200
	STDR-EXP	**0.0117**	**0.1596**	**0.0108**	**0.0132**	**0.0189**	**0.9241**	**0.9233**

TABLE II

The results of Mann–Whitney U tests.

Comparsion pairs	TDR-MG TDR-EXP	TDR-MG STDR-MG	TDR-MG STDR-EXP	TDR-EXP STDR-MG	TDR-EXP STDR-EXP
P-value	0.04448	0.0001230	0.0001649	0.0002198	0.0002198

TABLE III

Results comparisons with existing models.The best result is highlighted in bold.

Methods	10-th order Narma	20-th order Narma	Santa Fe Set-A	Santa Fe Set-D	Nonlinear Audio	Chlorine Concentration	Spoken Digit Recognition	M-W U test
	NMSE					Accuracy		P value
ESN	0.0762	0.6343	0.0579	0.0358	0.0538	0.6589	0.8133	3.527e-7
LSTM	0.1399	0.2759	0.0235	0.0241	0.0461	0.6880	0.9000	1.909e-7
RNN	0.1423	0.4292	0.0694	0.0287	0.0746	0.6312	0.8800	3.397e-8
mLSTM_FIXD	0.2067	0.3752	0.0277	0.0303	0.0308	0.6239	0.9010	3.397e-8
mRNN_FIXD	0.0189	0.2455	**0.0098**	0.0533	0.0454	0.6203	0.9000	0.0402
STDR-EXP	**0.0143**	**0.1804**	0.0110	**0.0170**	**0.0156**	**0.9164**	**0.9233**	-

improves predictive performance across tasks is STDR. EXP provides some help to improve predictive performance, but its main role is that of reducing the number of hyperparameters that need to be tuned while at least maintaining the predictive performance that would have been obtained with a tuned MG function.

2) Results comparisons with existing models: Besides the comparisons with standard TDR, we also compared our method with other existing models, which are vanilla ESN, vanilla LSTM, vanilla RNN, mLSTM_FIXD and mRNN_FIXD. The comparison results are shown in Table III. To make sure the fair comparison, the nodes size of our proposed STDR-EXP is set as the same as the hidden size of other models, which are 200 nodes.

In summary, our proposed method outperforms the existing models on both of the artificial datasets and real-world datasets for the prediction and classification tasks, except for the Santa Fe Set-A, where the proposed approach performed closely to mRNN_FIXD. The Mann–Whitney U tests of the existing models with STDR-EXP are conducted, of which P value are given in Table III. The level of significance is 0.05, therefore, we can confirm that our proposed STDR-EXP can improve the predictive performance and classification accuracy compared with the existing models.

V. Conclusion

In this paper, we have considered the problems of limited predictive performance and memory capacity, and hyperparameter-dependent oscillation function in standard TDRs, analyzing these limitations and giving their corresponding reasons. In order to address these limitations, a second-order time delay reservoir is proposed, which makes the time multiplexing of TDR more flexible to process time series. Specially, an oscillation function with exponential decay is applied to design the novel reservoir by DDE model, decoupling the dependency between related hyperparameter of reservoir and nonlinearity of data sets. The results of our experiments with multi-view data sets show that the second order time-delay reservoir outperform the standard TDR and some existing models on both of time series prediction and recognition tasks.

The proposed reservoir is limited to second order and even higher orders may be necessary for certain problems. In the future, we will study higher-order TDRs and make use of evolutionary approaches to search for an optimal order of TDR for different applied tasks. In addition, the computation cost of higher-order TDR will also be further studied.

REFERENCES

[1] M. Lukoševičius and H. Jaeger, “Reservoir computing approaches to recurrent neural network training,” *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.

[2] H. 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*, vol. 148, no. 34, p. 13, 2001.

[3] W. Maass, T. Natschläger, and H. Markram, “Real-time computing without stable states: A new framework for neural computation based on perturbations,” *Neural computation*, vol. 14, no. 11, pp. 2531–2560, 2002.

[4] G. Trierweiler Ribeiro, J. Guilherme Sauer, N. Fraccanabbia, V. Cocco Mariani, and L. dos Santos Coelho, “Bayesian optimized echo state network applied to short-term load forecasting,” *Energies*, vol. 13, no. 9, p. 2390, 2020.

[5] M. Mansoor, F. Grimaccia, S. Leva, and M. Mussetta, “Comparison of echo state network and feed-forward neural networks in electrical load forecasting for demand response programs,” *Mathematics and Computers in Simulation*, vol. 184, pp. 282–293, 2021.

[6] H. Hu, L. Wang, and S.-X. Lv, “Forecasting energy consumption and wind power generation using deep echo state network,” *Renewable Energy*, vol. 154, pp. 598–613, 2020.

[7] E. López, C. Valle, H. Allende, E. Gil, and H. Madsen, “Wind power forecasting based on echo state networks and long short-term memory,” *Energies*, vol. 11, no. 3, p. 526, 2018.

[8] G. T. Ribeiro, A. A. P. Santos, V. C. Mariani, and L. dos Santos Coelho, “Novel hybrid model based on echo state neural network applied to the prediction of stock price return volatility,” *Expert Systems with Applications*, vol. 184, p. 115490, 2021.

[9] C. Merkel, “Design of a time delay reservoir using stochastic logic: A feasibility study,” in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 2186–2192.

[10] K. Dhireesha, S. Qutaiba, M. Cory, T. James, and W. Bryant, “Design and analysis of a neuromemristive reservoir computing architecture for biosignal processing,” *Frontiers in Neuroscience*, vol. 9, p. 502, 2016.

[11] Y. Paquot, F. Duport, A. Smerieri, J. Dambre, B. Schrauwen, M. Haelterman, and S. Massar, “Optoelectronic reservoir computing,” *Scientific reports*, vol. 2, p. 287, 2012.

[12] L. Appeltant, M. C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, “Information processing using a single dynamical node as complex system,” *Nature communications*, vol. 2, no. 1, pp. 1–6, 2011.

[13] L. Grigoryeva, J. Henriques, L. Larger, and J.-P. Ortega, “Stochastic nonlinear time series forecasting using time-delay reservoir computers: Performance and universality,” *Neural Networks*, vol. 55, pp. 59–71, 2014.

[14] L. Keuninckx, J. Danckaert, and G. Van der Sande, “Real-time audio processing with a cascade of discrete-time delay line-based reservoir computers,” *Cognitive Computation*, vol. 9, no. 3, pp. 315–326, 2017.

[15] L. Loomis, N. McDonald, and C. Merkel, “An fpga implementation of a time delay reservoir using stochastic logic,” *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 14, no. 4, pp. 1–15, 2018.

[16] D. Brunner, M. C. Soriano, C. R. Mirasso, and I. Fischer, “Parallel photonic information processing at gigabyte per second data rates using transient states,” *Nature communications*, vol. 4, no. 1, pp. 1–7, 2013.

[17] J. Moon, W. Ma, J. H. Shin, F. Cai, C. Du, S. H. Lee, and W. D. Lu, “Temporal data classification and forecasting using a memristor-based reservoir computing system,” *Nature Electronics*, vol. 2, no. 10, pp. 480–487, 2019.

[18] F. Köster, D. Ehlert, and K. Lüdge, “Limitations of the recall capabilities in delay-based reservoir computing systems,” *Cognitive Computation*, pp. 1–8, 2020.

[19] A. G. Parlos, O. T. Rais, and A. F. Atiya, “Multi-step-ahead prediction using dynamic recurrent neural networks,” *Neural networks*, vol. 13, no. 7, pp. 765–786, 2000.

[20] M. C. Mackey and L. Glass, “Oscillation and chaos in physiological control systems,” *Science*, vol. 197, no. 4300, pp. 287–289, 1977.

[21] H. Sak, A. W. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” 2014.

[22] C. L. Giles, G. M. Kuhn, and R. J. Williams, “Dynamic recurrent neural networks: Theory and applications,” *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 153–156, 1994.

[23] J. Zhao, F. Huang, J. Lv, Y. Duan, Z. Qin, G. Li, and G. Tian, “Do rnn and lstm have long memory?” in *International Conference on Machine Learning*. PMLR, 2020, pp. 11 365–11 375.

[24] A. S. Weigend, *Time series prediction: forecasting the future and understanding the past*. Routledge, 2018.

[25] D. Verstraeten, J. Dambre, X. Dutoit, and B. Schrauwen, “Memory versus non-linearity in reservoirs,” in *The 2010 international joint conference on neural networks (IJCNN)*. IEEE, 2010, pp. 1–8.

[26] H. Jaeger, *Short term memory in echo state networks*. GMD-Forschungszentrum Informationstechnik, 2001, vol. 5.

[27] A. Rodan and P. Tiňo, “Simple deterministically constructed cycle reservoirs with regular jumps,” *Neural computation*, vol. 24, no. 7, pp. 1822–1852, 2012.