entropy

MDPI

Article

Reviving the Dynamics of Attacked Reservoir Computers

Ruizhi Cao [1], Chun Guan [1,*], Zhongxue Gan [1,*] and Siyang Leng [1,2]

[1] Institute of AI and Robotics, Academy for Engineering and Technology, Fudan University, Shanghai 200433, China
[2] Research Institute of Intelligent Complex Systems, Fudan University, Shanghai 200433, China
* Correspondence: chunguan@fudan.edu.cn (C.G.); ganzhongxue@fudan.edu.cn (Z.G.)

Abstract: Physically implemented neural networks are subject to external perturbations and internal variations. Existing works focus on the adversarial attacks but seldom consider attack on the network structure and the corresponding recovery method. Inspired by the biological neural compensation mechanism and the neuromodulation technique in clinical practice, we propose a novel framework of reviving attacked reservoir computers, consisting of several strategies direct at different types of attacks on structure by adjusting only a minor fraction of edges in the reservoir. Numerical experiments demonstrate the efficacy and broad applicability of the framework and reveal inspiring insights into the mechanisms. This work provides a vehicle to improve the robustness of reservoir computers and can be generalized to broader types of neural networks.

Keywords: reservoir computer; attack and recovery; Echo State Property; network structure

Citation: Cao, R.; Guan, C.; Gan, Z.; Leng, S. Reviving the Dynamics of Attacked Reservoir Computers. *Entropy* **2023**, *25*, 515. https://doi.org/10.3390/e25030515

Academic Editors: Jaroslaw Krzywanski, Yunfei Gao, Marcin Sosnowski, Karolina Grabowska, Dorian Skrobek, Ghulam Moeen Uddin, Anna Kulakowska, Anna Zylka and Bachil El Fil

Received: 2 February 2023
Revised: 8 March 2023
Accepted: 14 March 2023
Published: 16 March 2023

1. Introduction

Neural networks (NN) are subject to external perturbations and internal variations, especially when they are physically implemented [1–4]. In the past few decades, tremendous efforts have been devoted to relieving small perturbations on the input, that is, adversarial attacks [5–7], but are seldom made to consider the attack on the network structure. In fact, failure of certain neurons and/or synaptic connections may also significantly reduce the computational capacity [8,9], while biological NN compensate for this loss by adaptively adjusting/rebirthing links [10–12]. In clinical practice, neuromodulation techniques, such as the Transcranial Direct Current Stimulation (tDCS) [13–15], recover the neural functions through compensating neural connections with weak direct currents [16–18]. Restoring the network performance as much as possible from attacks on network structure becomes an urgent challenge. Albert et al. investigated the error tolerance and attack vulnerability under the removal of nodes in complex networks [19]. Nguyen et al. measured the network properties when a real-world complex network is attacked by different strategies [20]. Further studies concentrated on the recovery approaches [21,22], which are vital to the smooth functioning of the networked systems. However, the structural attack and recovery of neural networks are barely visited and require intensive investigation [23].

Reservoir Computer (RC), a variant of the Recurrent Neural Network (RNN), has enjoyed recent attention since the seminal works by Jaeger [24] and Maass [25], due to its excellent training efficiency and its convenience to be physically implemented [26]. Its architecture, comprising of an input layer, a linear output layer, and a reservoir network consisting of dynamical neurons, well-simulates the mechanism of biological brains in a conceptual manner [27]. While the input matrix and the reservoir matrix are randomly generated and fixed, training the output matrix occupies the whole expense, which can be efficiently obtained by least-squares optimization [28], and avails RC to reduce the complex training of RNN's parameters to a linear regression problem [24]. The fixed weights also enable the reservoir layer to be created with a specific physical system [29]. Recent works analytically proved that a suitably trained RC is essentially a high-dimensional

embedding of the input dynamical system [24,30–34], as shown in Figure 1a. When its reservoir becomes the target of an attack, on nodes and/or links, the performance may drop significantly and can not revive without external intervention [Figure 1b]. Similar operations performed in Deep Neural Networks (DNN) are known as *dropout* [35] and *pruning* [36], which are usually regarded as training tricks, instead of attacks, to improve the performance due to the redundant structures in DNN. Practically, the term "attacked" can be also interpreted/replaced as "failed" in broader scenarios. In physically implemented devices of RC, digital components, such as field-programmable gate arrays or digital signal processors, are used for the reservoir layer and readout layer. The memristor [29], a new type of information processing device which has a memory of past voltages or currents, is recently used to boost the power efficiency of the hardware implementations of reservoir computing systems. In these circumstances, the failures can usually be caused by the sudden disconnection between memristors during prolonged operations [37] or/and environmental damages to the internal electrical components [38]. These circumstances can be regarded as "attacks" to the reservoir and thus require "recovery". Therefore, the study of attack and recovery for RC is not only at the theoretical level, but also has practical significance.

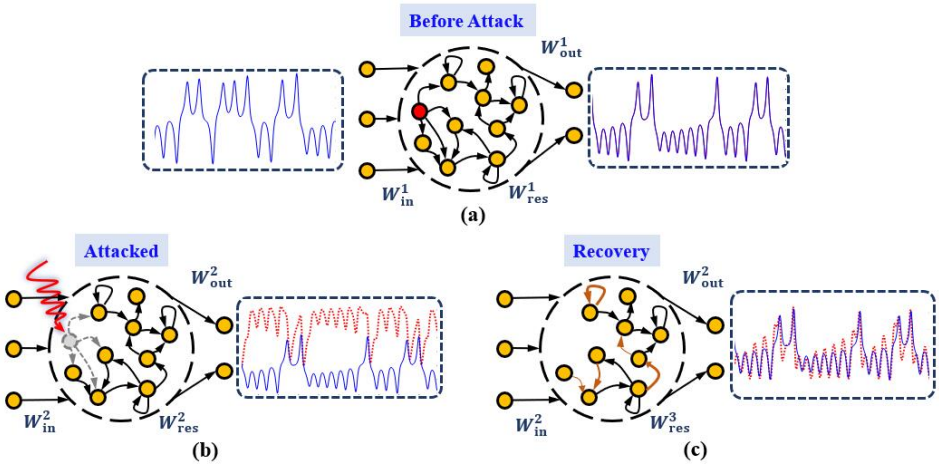

Figure 1. Schematic diagram of attack and recovery in reservoir computer. (**a**) Before attack, in the configuration of time series prediction, a standard RC accurately predicts the true dynamics, with blue and red lines denoting the true values and the predicted results respectively. (**b**) Attacked, here node attack is illustrated, failing its adjacent links (gray dashed arrows) and the predicted results deviate from the true values. (**c**) Recovery, by adjusting automatically part of the remaining links (orange arrows), the performance of RC improves significantly.

In this paper, analogous to the neural compensation mechanism in brains, we design several recovering mechanisms for reservoir computer to compensate its performance loss under different types of attacks on structure, that is, adjusting only a minor fraction of edges in the reservoir according to different attack scenarios. Results show that it is impossible to take precautions on specific nodes/links due to the ambiguous relationship between their *a priori* topological measurement and the performance loss under attack. Our proposed strategies successfully and efficiently revive the functioning of RC by automatically adjusting the remaining neurons/synapses [Figure 1c], which represent practical advancement towards enhancing the robustness of neural networks.

The paper is organized as follows: Section 2 reviews the standard reservoir computer and introduces the attack and recovery strategies employed in this study. Section 3 presents the performance loss under attack and the corresponding recovery results. We also quantitatively and systematically analyze the different recovery strategies. Section 4 discusses several important related issues and concludes the paper.

2. Method

2.1. Standard Reservoir Computer

The standard framework of RC can be described in the state updating rule of the reservoir neurons [24]:

$$\boldsymbol{r}_k = (1-\alpha)\boldsymbol{r}_{k-1} + \alpha\phi(W_{\text{res}}\boldsymbol{r}_{k-1} + W_{\text{in}}\boldsymbol{x}_k), \tag{1}$$

where $\boldsymbol{r}_k \in \mathbb{R}^m$ represents the state of m reservoir neurons at time step k, and $\boldsymbol{x}_k \in \mathbb{R}^n$ is the input signal observed from a dynamical system φ evolving on a compact manifold $\mathcal{M}$. $W_{\text{in}} \in \mathbb{R}^{m\times n}$ and $W_{\text{res}} \in \mathbb{R}^{m\times m}$ denote the input weight matrix and the reservoir network matrix respectively, which are randomly generated according to certain distribution laws and then fixed. We consider two settings, full connection and sparse connection, of the reservoir, while in the latter case two nodes i, j are called linked if $W_{\text{res}}^{ij} \neq 0$ or $W_{\text{res}}^{ji} \neq 0$ (W_{res}^{ij} denotes the link weight from node i to node j). $\alpha \in (0, 1)$ is the leakage factor controlling the time-scale mismatch between the input and reservoir dynamics ($\alpha = 1$ represents the previous states do not leak into the current states) and function ϕ determines the dynamics of the reservoir neurons which at its simplest can be set as $\tanh(\cdot)$. Consequently, the output $\boldsymbol{y}_k \in \mathbb{R}^l$ linearly combines the reservoir states such that $\boldsymbol{y}_k = W_{\text{out}}\boldsymbol{r}_k$, with the output weight matrix $W_{\text{out}} \in \mathbb{R}^{l\times m}$ solely requiring training. RC can be adapted to different tasks while in the task of one-step time series prediction [39–41], the target is $\hat{\boldsymbol{y}}_k := \boldsymbol{x}_{k+1}$ and W_{out} can be calculated by minimizing the loss function

$$\mathcal{L} = \sum_{k=1}^{N} ||\boldsymbol{x}_{k+1} - W_{\text{out}}\boldsymbol{r}_k||^2 + \beta||W_{\text{out}}||^2, \tag{2}$$

where $\beta > 0$ is the L_2-regularization coefficient. After the training phase, the output $\boldsymbol{y}_k$ can be redirected to the input layer, that is, $\boldsymbol{x}_{k+1} := \boldsymbol{y}_k$, thus RC runs in an autonomous mode (in this case $l = n$). With these settings, RC is proved to intactly capture the dynamics of the input dynamical system, which naturally requires the reservoir's initial state to fade away, that is, the Echo State Property (ESP) [24,42]. A sufficient condition guaranteeing the ESP is that the spectral radius of W_{res} is smaller than 1, that is, $||W_{\text{res}}|| < 1$. Thus in this study we rescale the reservoir network matrix by

$$W_{\text{res}} := \frac{\rho^*}{\rho(W_{\text{res}})} W_{\text{res}}, \tag{3}$$

where $\rho(\cdot)$ denotes the spectral radius or the maximum eigenvalue of a matrix, and ρ^* is the desired spectral radius.

We use the Mean Squared Error (MSE) to evaluate RC's performance at different stages:

$$\text{MSE}_k := \frac{1}{\tau} \sum_{p=k-\tau+1}^{k} ||\boldsymbol{y}_p - \hat{\boldsymbol{y}}_p||, \tag{4}$$

where τ denotes a specified time window. Notice that a well-trained RC can achieve accurate prediction with saving computations [Figure 2a,b].

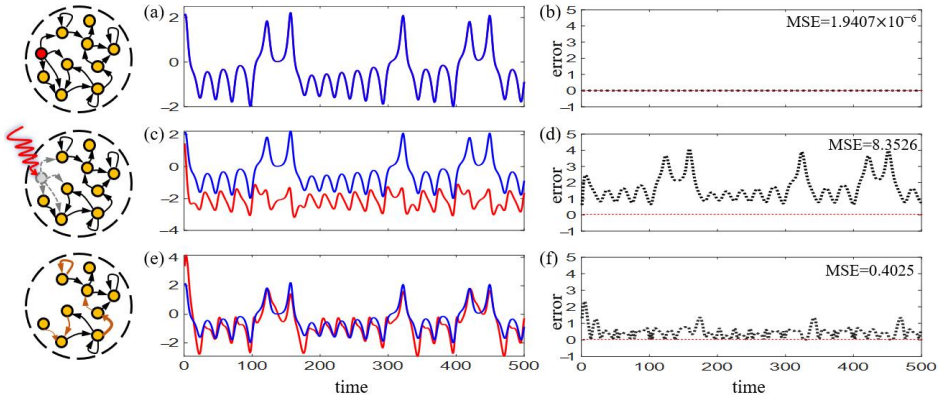

Figure 2. Node-attack and recovery for fully connected RC. (**a**) Before attack, predicted (**red**) and true (**blue**) time series. Here the two trajectories coincide with each other. (**c**) Attacked, predicted (**red**) and true (**blue**) time series. (**e**) Recovery, predicted (**red**) and true (**blue**) time series. (**b**,**d**,**f**) The prediction errors for each corresponding stage. Here the MSE value denotes an average of 50 realizations.

2.2. Reservoir-Attack Mechanisms

RC runs in an autonomous mode once the output matrix W_{out} is trained and fixed. We propose two possible attack mechanisms on the reservoir network, that is, node-attack and edge-attack, while in biological neural networks, the two types of attack may represent the apoptosis of neurons and the fracture of synapses respectively. In physical implementations, the two types correspond to failure of a single memristor and disconnection of the circuit between memristors respectively. It is shown that RC may lose efficacy under both attack mechanisms [Figure 2c,d].

Mechanism 1 (Node-attack). *For a well-trained and autonomous RC, node-attack denotes the removal of certain node s and its all adjacent edges, which is performed by*

$$\boldsymbol{r}_k := \mathbf{0}, \quad W_{\text{res}}^{:,s} := 0, \quad W_{\text{res}}^{s,:} := 0, \quad W_{\text{in}}^{:,s} := 0, \quad W_{\text{out}}^{s,:} := 0,$$

where $k > k^*$, k^* is the attack time, and the superscript ":" denotes the corresponding row/column.

Mechanism 2 (Edge-attack). *For a well-trained and autonomous RC, edge-attack denotes the removal of certain link from node i to node j, which is performed by*

$$W_{\text{res}}^{ij} := 0.$$

In practice, the attack can be launched to a proportion of the nodes and/or edges according to certain rule.

2.3. Rc-Revive Strategies

We denote W_{res} and $\tilde{W}_{\text{res}}$ as the reservoir matrix before and after attack respectively. To realize recovery in an energy-efficient manner, we revive RC's performance by adjusting only a small fraction of the weights in $\tilde{W}_{\text{res}}$, which are treated as values to be optimized to achieve minimal MSE, leading to the revived reservoir matrix W_{res}^*.

Different optimization methods can be utilized to achieve the goal. In this study, Simulated Annealing (SA) [43] is used and integrated in our proposed reviving strategies to automatically find the optimal set of connections to impose adjustment, with the specific procedures presented in Algorithm 1.

Algorithm 1 Simulated Annealing-based recovery of RC

Input: $\tilde{W}_{\text{res}}$, $\mathcal{E}$ is a set of edges in $\tilde{W}_{\text{res}}$ allowing adjusting depending on strategies, $k_{\max}$ is the maximum iterations, N is the number of edges perturbed each time during SA
Output: W^*_{res}

```
MSE = MSE(W̃_res)
for t = 1 → k_max do
    Perturb N edges randomly from E by adding random(−1, 1) to obtain W*_res
    MSE* = MSE(W*_res)
    if MSE* < MSE then
        P = 1
    else
        P = exp(−(MSE* − MSE) / 0.95^t)
    end if
    if P ⩾ random(0, 1) then
        MSE = MSE*, W̃_res = W*_res
    end if
end for
W*_res = W̃_res
```

We first propose two strategies to revive RC from node-attack. Here the reservoir structure is preserved during the recovery process, that is, no new links is allowed to generate [Figure 1c]. As shown in the following Strategy 1 and 2, we choose the set of connections in a completely random manner or related to the attacked node respectively. In physical implementations, the selected set denotes partial connections between the memristors.

Strategy 1 (**Full selection**). *The allowing set of edges here is defined as*

$$\mathcal{E}_{\text{F}} := \{(i,j)|i,j \neq s, \tilde{W}^{ij}_{\text{res}} \neq 0\}.$$

Strategy 2 (**Relevant selection**). *The allowing set of edges here is defined as*

$$\mathcal{E}_{\text{R}} := \{(i,j)|i,j \neq s, \tilde{W}^{ij}_{\text{res}} \neq 0, W^{is}_{\text{res}}, W^{si}_{\text{res}}, W^{js}_{\text{res}}, W^{sj}_{\text{res}} \text{ are not all zero}\}.$$

In this study, we consider both fully and sparsely connected reservoir structures, while Strategy 1 and 2 become the same for the former case and we compare the two strategies for the latter case. Biologically, the relevant selection strategy is more ubiquitous since compensation always occurs at the neighbouring neurons [10–12].

We next consider the recovery of edge-attack. The above two strategies can be utilized in an analogous manner and achieve good recovery results. Here we propose another strategy allowing adding new links, which enriches the structure of the reservoir and represents the birth of new synapses. This strategy is more natural in physiology, but difficult to implement in digital devices.

Strategy 3 (**Incremental selection**). *The allowing set of edges here contains all rest of the edges:*

$$\mathcal{E}_{\text{I}} := \{(i,j)|(i,j) \text{ is not attacked}\}.$$

We trade off the selected edges by γ percentage of existing edges and $1-\gamma$ percentage of adding edges during SA.

Note that retraining the RC may be considered as a solution. However, considering practical scenarios, retraining usually requires re-collecting a large amount of training data. Moreover, the operators of the devices often are not authorized to touch the system underpinnings [44], which imposes a requirement of adaptive recovery mechanism that does not need external interventions.

3. Result

We test and analyze the proposed methods in reservoir computers with $m = 100$ reservoir neurons and leaky factor $\alpha = 0.25$, which is trained with the normalized x component of the benchmark Lorenz system:

$$\begin{cases} \dot{x} = -\sigma x + \sigma y, \\ \dot{y} = rx - y - xz, \\ \dot{z} = -bz + xy \end{cases} \tag{5}$$

with $\sigma = 10, r = 28, b = 8/3$ [45]. 1000 points with discretization time step 0.03 are used as training set, and the leading 100 reservoir states are discarded to eliminate the transient behavior. The input matrix W_{in} and the reservoir matrix W_{res} are generated with the elements randomly selected from $[-1, 1]$. W_{res} is rescaled to a spectral radius of 0.9. We set the initial value of the SA procedure to be the reservoir state after attack, that is, $\tilde{W}_{\text{res}}$, and update the temperature every 100 iterations with $k_{\max} = 20{,}000$ maximum iterations.

3.1. Node-Attack and Recovery

Before attack, the RC is well-trained to a high prediction accuracy, see Figure 2a,b, while it loses efficacy when one randomly selected node is attacked [Figure 2c,d]. Strategy 1 is utilized with $N = 50$ to fully connected RC and achieves good recovery result that significantly reducing the prediction errors [Figure 2e,f]. Notice that for fully connected RC, performing Strategy 2 is completely equivalent.

For sparsely connected RC (here we allow 10% connections), both strategies are applied which successfully revive the performance to satisfactory extent [Figure 3]. However, Strategy 1 obtains relatively higher MSE (0.6836, Figure 3d) than the relevant selection strategy (0.3450, Figure 3c). Possible explanation lies in that adjacent links can be regarded as belonging to the same subgraph containing the attacked node, thus share similar local information, facilitating better recovery by adjusting them [46].

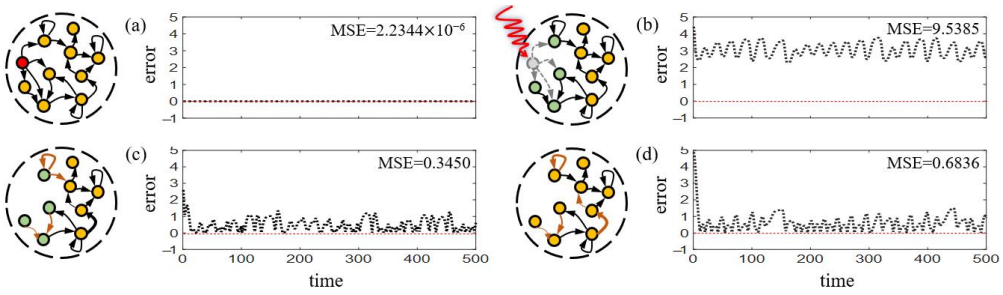

Figure 3. Node-attack and recovery for sparsely connected RC. The prediction errors for stages of (**a**) before attack, (**b**) attacked, (**c**) recovery with relevant selection, (**d**) recovery with full selection, respectively. Green nodes denote the neighbours of the attacked node. Here the MSE value denotes an average of 50 realizations.

To demonstrate the mechanisms can be applied to various tasks, we additionally consider a system reconstructing task of the Rössler system:

$$\begin{cases} \dot{x} = -\omega y - z, \\ \dot{y} = \omega x + \alpha y, \\ \dot{z} = \beta + z(x - \gamma) \end{cases} \tag{6}$$

with $\omega = 1, \alpha = 0.2, \beta = 0.4$, and $\gamma = 5.7$. Here normalized time series x_t and y_t are used to reconstruct the dynamics of z_t and the parameter settings of RC are the same with the above experiment. We impose a node-attack to a fully connected RC and use Strategy 1 for recovery with $N = 50$, and the experimental results are shown in Figure 4. Notice that after the attack, RC's reconstructing ability is significantly weakened, with large MSE of 74.45.

After taking the strategy, the reconstructing successfully recovers with the MSE decreasing to 12.75.

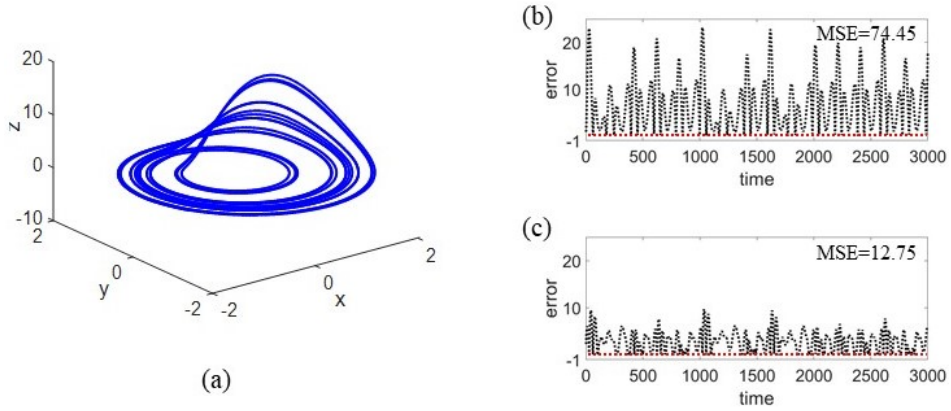

Figure 4. Node-attack and recovery for fully connected RC in the task of reconstructing Rössler system. (**a**) The original Rössler attractor. The reconstructing errors for stages of (**b**) attacked and (**c**) recovery are shown. Here the MSE value denotes an average of 50 realizations.

3.2. Edge-Attack and Recovery

Here attack is launched to 10% randomly selected edges. We apply Strategy 3 with $\gamma = 1$ and $N = 50$ to a fully connected RC and present the results in Figure 5a–c, which also reduces the prediction errors significantly and revives the RC to a healthy condition. Notice that for fully connected RC, $\gamma < 1$ leads to the rebirth of the attacked edges, which is not allowed in our settings. While for sparsely connected RC, we analyze the recovery effects with different choices of γ. Notice that $\gamma = 0$ denotes the circumstance that recovery is reached only by generating new edges and preserving all the existing ones, which produces the MSE of 0.3080 [Figure 6b]. When γ increases to 0.5, denoting a mixing strategy of generating new edges and adjusting existing edges, recovery MSE also increases to 0.4551 [Figure 6c]. Additionally, the strategy with $\gamma = 1$ produces the highest MSE of 0.5896 [Figure 6d]. The above MSE values are obtained through averaging the results of 50 realizations. The experiments show that in a sparsely connected RC, adding new edges becomes the best recovery strategy, which represents an enrichment of the reservoir's structure. This is in accordance with biological nervous system, in which compensation is always reached by generating new synapses [10–12].

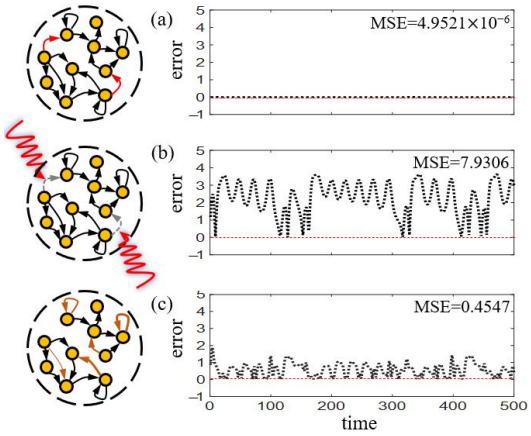

Figure 5. Edge-attack and recovery for fully connected RC. The prediction errors for stages of (**a**) before attack, (**b**) attacked, (**c**) recovery, respectively. Here two edges are attacked for illustration. Here the MSE value denotes an average of 50 realizations.

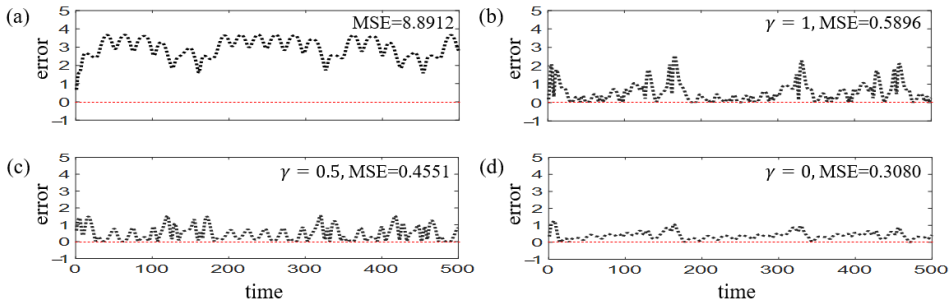

Figure 6. Edge-attack and recovery for sparsely connected RC. The prediction errors for stages of (**a**) attacked, (**b**) recovery with Strategy 3 ($\gamma = 1$), (**c**) recovery with Strategy 3 ($\gamma = 0.5$), (**d**) recovery with Strategy 3 ($\gamma = 0$), respectively. Here the MSE value denotes an average of 50 realizations.

4. Discussion

4.1. Ineffectiveness of Precaution to Reservoir

To avoid performance collapsing under attack, precautionary actions may be taken. For example, if we can identify the most essential nodes/edges in advance, protections can be imposed to these targets. Usually, there are many methods/measurements from complex network theory to evaluate the importance of nodes based on their local or global information, e.g., Degree Centrality [47], Node Strength [48], Betweenness Centrality [49], PageRank [50], and so forth. For a fully connected reservoir, all nodes have indistinguishable evaluations, preventing the emergence of the key nodes. Here we show in the framework of sparse reservoir computer, the performance loss under node-attack is also irrelevant with node's importance measurements. We test the linear relationship between MSE under node-attack and several measurements of the node, and list the results in Table 1, demonstrating all nodes are of similar importance in the reservoir. Therefore, taking precautionary actions is difficult, and recovery strategies proposed in this work are of great significance. In addition, previous experiments show that although the reservoir structure is initially randomly chosen, the performance is quite sensitive to small perturbations in the network structure, further demonstrating the significance of this work.

Table 1. Relationship between node-attack MSE and importance measurements ($y = ax + b$).

Importance Measurement	a	R^2-Score
Degree Centrality	-0.32 ± 0.65	0.03
Node Strength	-0.51 ± 0.72	0.06
Betweenness Centrality	$+0.28 \pm 0.52$	0.12
PageRank	-0.30 ± 0.81	0.07

4.2. Echo State Property vs. Attack and Recovery

The spectral radius of the reservoir matrix plays crucial role in determining the performance of RC by ensuring the Echo State Property and balancing memory capacity and nonlinearity [51]. We are interested in the variation of the spectral radius, thus the ESP, during attack and recovery. A rigorous theory characterizing this variation lies in the spectral graph theory and can be referred to reference [52]. Here we perform numerical analysis using 100×100 randomly and sparsely connected reservoir network matrix rescaled to different initial spectral radius and randomly remove an increasing proportion edges. As shown in Figure 7a, the spectral radius has a decreasing trend following the removing procedure, harming the performance of RC, as demonstrated in [24] that RC runs most effectively with a spectral radius close to 1. However the ESP remains satisfied.

For the recovery, we successively added new edges to the reservoir network, and present the results in Figure 7b. We find the spectral radius gradually increases as the network gets denser, but will exceed 1 eventually, violating the ESP and failing the performance. However, in our framework, utilizing Strategy 3 together with the Simulated

Annealing optimization produces Figure 7c, which shows that the spectral radius stabilizes at around 1 and the network stops growing automatically, guaranteeing the best performance of recovery.

Meanwhile, we experimentally verify that whether the recovery can be achieved by adjusting the spectral radius. We rescale the fully connected reservoir matrix being node-attack, and find that the MSE shows a slow decreasing trend with the turning up of the spectral radius (still remains large after adjustment, see Figure 8). Nevertheless, adjusting the spectral radius requires an overall manipulation of the whole matrix, which is time-consuming compared to our proposed strategies.

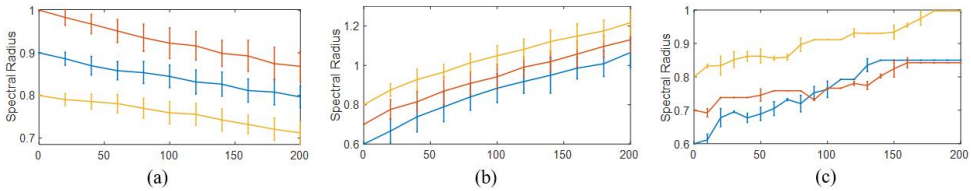

Figure 7. Variation of spectral radius during attack and recovery. (**a**) A decreasing trend from differently selected initial values of the spectral radius during edge-attack, harming the performance of RC. (**b**) The spectral radius gradually increases as the network gets denser, but will exceed 1 eventually, violating the ESP and failing the performance. (**c**) With our framework, the spectral radius stabilizes at around 1 and the network stops growing automatically, guaranteeing the best performance of recovery. The horizontal axis show the numbers of added/removed edges. Error bar denotes standard deviation of 20 realizations.

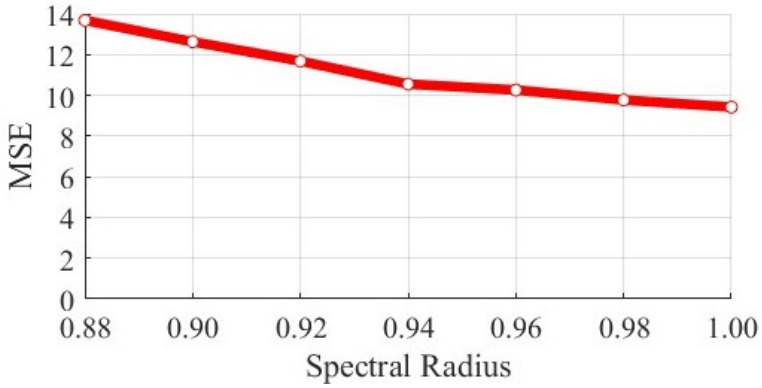

Figure 8. Variation of MSE during spectral radius adjustment. MSE gradually decreases with the increase in the spectral radius, but the recovery is inefficient and the MSE after recovery remains large compared to our proposed strategies.

4.3. How to Choose Recovery Strategies

Exposing to different attacked scenarios, we analyze the proper selection of recovery strategies. Two criteria, reservoir connectedness [53] and attacked edge betweenness centrality [54], are adopted to determine the optimal strategy. Here we compare two attacked scenarios for a sparsely connected RC, with its healthy state presented in Figure 2a. The first case (attack 90% connections) renders the reservoir separating into several connected components, significantly harming its prediction ability [Figure 9a]. Strategy 2 is applied with s adjusting to be the vertexes of the attacked edges, decreasing the MSE to 0.7182 [Figure 9b]. As comparison, Strategy 3 with $\gamma = 0$ is applied and achieves generally better recovery results. At optimal case that the reservoir returns to be connected, smaller MSE of 0.4976 is achieved [Figure 9c]. However, for the second case of attack that does not harm the reservoir's connectedness, the two strategies have similar results, with MSE values of 0.8979 and 0.8176 respectively.

These results demonstrate the importance of the reservoir connectedness in selecting optimal strategies. Generally, against massive attacks, adding new edges to restore the

reservoir connectedness can be preferentially chosen, as connectedness plays crucial role in ensuring the network's proper functioning [53].

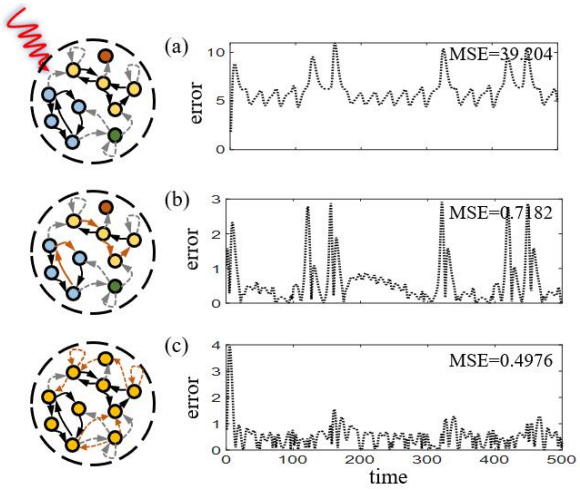

Figure 9. Different choices of recovery strategies according to the reservoir connectedness. The prediction errors for stages of (**a**) attacked, (**b**) recovery with Strategy 2, (**c**) recovery with Strategy 3 ($\gamma = 0$), respectively. Here different colors of the nodes denote their connectedness and the MSE value denotes an average of 50 realizations.

With the same configurations, we compare Strategy 2 and Strategy 3 respectively in two attacked scenarios with the average betweenness of the attacked edges greater (3.8119) or smaller (3.0427) than the average betweenness of the reservoir before attack (3.414). Results in Figure 10 show the preference of Strategy 3 when the attacked edges are relatively important and indiscriminate selection of Strategy 2 and 3 for less important attacked edges. In fact, a higher value of the betweenness implies the criticality of the edges [55], and adding new edges avails recovery when crucial edges are attacked.

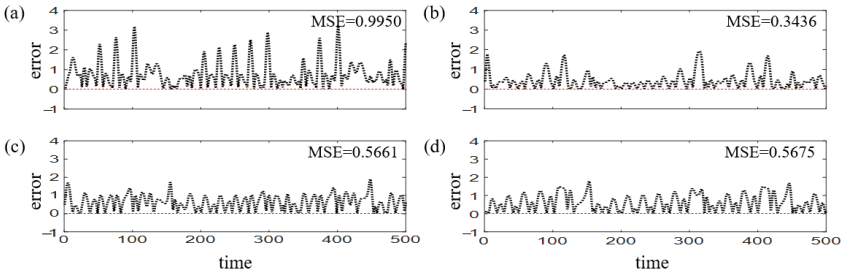

Figure 10. Different choices of recovery strategies according to the attacked edge betweenness centrality. When the betweenness of the attacked edges is high, the prediction errors for stages of (**a**) recovery with Strategy 2, (**b**) recovery with Strategy 3. When the betweenness of the attacked edges is low, the prediction errors for stages of (**c**) recovery with Strategy 2, (**d**) recovery with Strategy 3. Here the MSE value denotes an average of 50 realizations.

4.4. Minor Adjusting Is Sufficient for Recovery

In practice, efficient recovery is expected to achieve at minimized cost, which in our study is reflected in the number of edges perturbed (N) during optimization, yielding a trade-off between performance and cost. Here we search for an optimized N for a fully connected RC under node-attack with full selection recovery strategy. The experimental configurations are the same with Figure 2, but varying N from 10 to 100. As compared to the network size (100×100), recovery can be reached with extremely minor adjusting and low cost. As shown in Figure 11, lowest MSE is reached at $N = 50$ and increasing the number of adjusted edges does not benefit the efficacy of recovery. This result further demonstrates the broad applicability and high efficiency of our proposed framework.

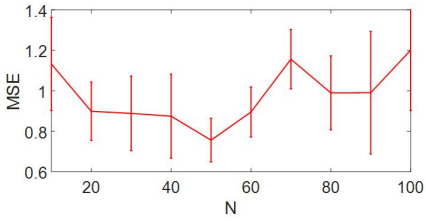

Figure 11. Efficacy of recovery v.s. number of adjusted edges for fully connected RC under node-attack. Lowest MSE is reached at $N = 50$, demonstrating that minor adjusting is sufficient for recovery. Error bar denotes standard deviation of 20 realizations.

4.5. Attack and Recovery in Other Neural Networks

Reservoir computing, as a specific variant of RNN, is subject to not only attacks on network structure but also adversarial attacks, while the latter is more commonly considered in traditional NN, e.g., Fast Gradient Sign Method (FGSM) deteriorates the performance in tasks of time series prediction and graph node classification [56,57], DeepFake causes the face recognition model to misclassify [58–60], and so forth. Adversarial attacks on RC also represent a promising topic which will be included in our future work, including on other variants of RNN, such as LSTM [61] and GRU [62]. Moreover, attacks on network structure of Deep Neural Networks (DNN) are regarded as training tricks, instead of attacks, to improve the performance due to the redundant structures in DNN.

5. Conclusions

In this paper, inspired by the biological neural compensation mechanism in brains, we proposed a framework of reviving attacked reservoir computers, consisting of several strategies directed to different types of attacks. All the strategies achieved sound recovery results. The analysis further brings inspiring insights, that:

(1) Adjusting adjacent neurons/synapses is more effective than distant ones;
(2) Enriching the reservoir network is more effective than adjusting existing edges;
(3) Reservoir connectedness and attacked edge betweenness centrality are crucial criteria in choosing optimal recovery strategies; and
(4) Minor adjustments are sufficient for recovery.

Future work includes incorporating advanced optimization algorithms, theoretical analysis on the choice of adjusting connections, and designing more adaptive strategies. The proposed attack and recovery strategies can be generalized to more variants of RNN, including LSTM and GRU. This work provided a practical framework to improve the robustness of reservoir computers, and a vehicle towards broader types of neural networks.

Author Contributions: Conceptualization, S.L.; Funding acquisition, Z.G. and S.L.; Investigation, R.C. and C.G.; Methodology, R.C., C.G. and S.L.; Supervision, Z.G. and S.L.; Validation, R.C.; Writing—original draft, R.C.; Writing—review & editing, R.C., C.G., Z.G. and S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the STI 2030—Major Projects (No. 2021ZD0201301), the National Natural Science Foundation of China (No. 12101133), and Shanghai Sailing Program (No. 21YF1402300). This work was also supported by Shanghai Municipal Science and Technology Major Project (No. 2021SHZDZX0103).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yu, F.; Qin, Z.; Liu, C.; Zhao, L.; Wang, Y.; Chen, X. Interpreting and Evaluating Neural Network Robustness. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, International Joint Conferences on Artificial Intelligence Organization, Macao, China, 10–16 August 2019; pp. 4199–4205. [CrossRef]
2. Huang, X.; Kroening, D.; Ruan, W.; Sharp, J.; Sun, Y.; Thamo, E.; Wu, M.; Yi, X. A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability. *Comput. Sci. Rev.* **2020**, *37*, 100270. [CrossRef]
3. Su, J.; Vargas, D.V.; Sakurai, K. One Pixel Attack for Fooling Deep Neural Networks. *IEEE Trans. Evol. Comput.* **2019**, *23*, 828–841. [CrossRef]
4. Draghici, S. Neural networks in analog hardware—Design and implementation issues. *Int. J. Neural Syst.* **2000**, *10*, 19–42. [CrossRef] [PubMed]
5. Fawaz, H.I.; Forestier, G.; Weber, J.; Idoumghar, L.; Muller, P.A. Adversarial attacks on deep neural networks for time series classification. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–8. [CrossRef]
6. Karim, F.; Majumdar, S.; Darabi, H. Adversarial attacks on time series. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 3309–3320. [CrossRef]
7. Jin, W.; Li, Y.; Xu, H.; Wang, Y.; Ji, S.; Aggarwal, C.; Tang, J. Adversarial attacks and defenses on graphs. *ACM SIGKDD Explor. Newsl.* **2021**, *22*, 19–34. [CrossRef]
8. Cohen, R.; Erez, K.; Ben-Avraham, D.; Havlin, S. Breakdown of the internet under intentional attack. *Phys. Rev. Lett.* **2001**, *86*, 3682. [CrossRef]
9. Bellingeri, M.; Cassi, D.; Vincenzi, S. Efficiency of attack strategies on complex model and real-world networks. *Phys. A Stat. Mech. Its Appl.* **2014**, *414*, 174–180. [CrossRef]
10. Marder, E.; Goaillard, J.M. Variability, compensation and homeostasis in neuron and network function. *Nat. Rev. Neurosci.* **2006**, *7*, 563–574. [CrossRef]
11. Gehring, W.J.; Goss, B.; Coles, M.G.; Meyer, D.E.; Donchin, E. A neural system for error detection and compensation. *Psychol. Sci.* **1993**, *4*, 385–390. [CrossRef]
12. Song, J.; Birn, R.M.; Boly, M.; Meier, T.B.; Nair, V.A.; Meyerand, M.E.; Prabhakaran, V. Age-related reorganizational changes in modularity and functional connectivity of human brain networks. *Brain Connect.* **2014**, *4*, 662–676. [CrossRef]
13. Biou, E.; Cassoudesalle, H.; Cogné, M.; Sibon, I.; De Gabory, I.; Dehail, P.; Aupy, J.; Glize, B. Transcranial direct current stimulation in post-stroke aphasia rehabilitation: A systematic review. *Ann. Phys. Rehabil. Med.* **2019**, *62*, 104–121. [CrossRef]
14. Pelletier, S.J.; Cicchetti, F. Cellular and molecular mechanisms of action of transcranial direct current stimulation: Evidence from in vitro and in vivo models. *Int. J. Neuropsychopharmacol.* **2015**, *18*, pyu047.
15. DaSilva, A.F.; Volz, M.S.; Bikson, M.; Fregni, F. Electrode positioning and montage in transcranial direct current stimulation. *JoVE* **2011**, *51*, e2744. [CrossRef]
16. Mancini, M.; Brignani, D.; Conforto, S.; Mauri, P.; Miniussi, C.; Pellicciari, M.C. Assessing cortical synchronization during transcranial direct current stimulation: A graph-theoretical analysis. *NeuroImage* **2016**, *140*, 57–65. [CrossRef]
17. Brunoni, A.R.; Nitsche, M.A.; Bolognini, N.; Bikson, M.; Wagner, T.; Merabet, L.; Edwards, D.J.; Valero-Cabre, A.; Rotenberg, A.; Pascual-Leone, A.; et al. Clinical research with transcranial direct current stimulation (tDCS): Challenges and future directions. *Brain Stimul.* **2012**, *5*, 175–195. [CrossRef]
18. Nitsche, M.A.; Boggio, P.S.; Fregni, F.; Pascual-Leone, A. Treatment of depression with transcranial direct current stimulation (tDCS): A review. *Exp. Neurol.* **2009**, *219*, 14–19. [CrossRef]
19. Albert, R.; Jeong, H.; Barabási, A.L. Error and attack tolerance of complex networks. *Nature* **2000**, *406*, 378–382. [CrossRef]
20. Nguyen, Q.; Pham, H.D.; Cassi, D.; Bellingeri, M. Conditional attack strategy for real-world complex networks. *Phys. A Stat. Mech. Its Appl.* **2019**, *530*, 121561. [CrossRef]
21. Khunasaraphan, C.; Vanapipat, K.; Lursinsap, C. Weight shifting techniques for self-recovery neural networks. *IEEE Trans. Neural Netw.* **1994**, *5*, 651–658. [CrossRef]
22. Xu, Z.; Lin, M.; Liu, J.; Chen, J.; Shao, L.; Gao, Y.; Tian, Y.; Ji, R. Recu: Reviving the dead weights in binary neural networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 5198–5208. [CrossRef]
23. Sanhedrai, H.; Gao, J.; Bashan, A.; Schwartz, M.; Havlin, S.; Barzel, B. Reviving a failed network through microscopic interventions. *Nat. Phys.* **2022**, *18*, 338–349. [CrossRef]
24. Lukoševičius, M.; Jaeger, H. Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.* **2009**, *3*, 127–149. [CrossRef]
25. Maass, W.; Natschläger, T.; Markram, H. A model for real-time computation in generic neural microcircuits. *Adv. Neural Inf. Process. Syst.* **2002**, *15*.
26. Hadaeghi, F.; He, X.; Jaeger, H. *Unconventional Information Processing Systems, Novel Hardware: A Tour D'Horizon*; IRC-Library, Information Resource Center der Jacobs University Bremen: Bremen, Germany, 2017. [CrossRef]
27. Buonomano, D.V.; Maass, W. State-dependent computations: Spatiotemporal processing in cortical networks. *Nat. Rev. Neurosci.* **2009**, *10*, 113–125. [CrossRef] [PubMed]

28. Ren, B.; Ma, H. Global optimization of hyper-parameters in reservoir computing. *Electron. Res. Arch.* **2022**, *30*, 2719–2729. [CrossRef]
29. Zhong, Y.; Tang, J.; Li, X.; Liang, X.; Liu, Z.; Li, Y.; Xi, Y.; Yao, P.; Hao, Z.; Gao, B.; et al. A memristor-based analogue reservoir computing system for real-time and power-efficient signal processing. *Nat. Electron.* **2022**, *5*, 672–681. [CrossRef]
30. Leng, S.; Aihara, K. Common stochastic inputs induce neuronal transient synchronization with partial reset. *Neural Netw.* **2020**, *128*, 13–21. [CrossRef]
31. Hart, A.; Hook, J.; Dawes, J. Embedding and approximation theorems for echo state networks. *Neural Netw.* **2020**, *128*, 234–247. [CrossRef]
32. Jaeger, H.; Lukoševičius, M.; Popovici, D.; Siewert, U. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Netw.* **2007**, *20*, 335–352. [CrossRef]
33. Verstraeten, D.; Schrauwen, B.; Stroobandt, D. Reservoir-based techniques for speech recognition. In Proceedings of the 2006 IEEE International Joint Conference on Neural Network Proceedings, Vancouver, BC, Canada, 16–21 July 2006; IEEE: Piscataway, NJ, USA, 2006; pp. 1050–1053. [CrossRef]
34. Vlachas, P.R.; Pathak, J.; Hunt, B.R.; Sapsis, T.P.; Girvan, M.; Ott, E.; Koumoutsakos, P. Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. *Neural Netw.* **2020**, *126*, 191–217. [CrossRef]
35. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
36. Han, S.; Pool, J.; Tran, J.; Dally, W. Learning both weights and connections for efficient neural network. *Adv. Neural Inf. Process. Syst.* **2015**, *28* [CrossRef]
37. Liu, K.; Dang, B.; Zhang, T.; Yang, Z.; Bao, L.; Xu, L.; Cheng, C.; Huang, R.; Yang, Y. Multilayer Reservoir Computing Based on Ferroelectric α-In2Se3 for Hierarchical Information Processing. *Adv. Mater.* **2022**, *34*, 2108826. [CrossRef]
38. Liang, X.; Zhong, Y.; Tang, J.; Liu, Z.; Yao, P.; Sun, K.; Zhang, Q.; Gao, B.; Heidari, H.; Qian, H.; et al. Rotating neurons for all-analog implementation of cyclic reservoir computing. *Nat. Commun.* **2022**, *13*, 1549. [CrossRef]
39. Fan, H.; Jiang, J.; Zhang, C.; Wang, X.; Lai, Y.C. Long-term prediction of chaotic systems with machine learning. *Phys. Rev. Res.* **2020**, *2*, 012080. [CrossRef]
40. Ma, H.; Leng, S.; Aihara, K.; Lin, W.; Chen, L. Randomly distributed embedding making short-term high-dimensional data predictable. *Proc. Natl. Acad. Sci. USA* **2018**, *115*, E9994–E10002. [CrossRef]
41. Chen, P.; Liu, R.; Aihara, K.; Chen, L. Autoreservoir computing for multistep ahead prediction based on the spatiotemporal information transformation. *Nat. Commun.* **2020**, *11*, 4568. [CrossRef]
42. Yildiz, I.B.; Jaeger, H.; Kiebel, S.J. Re-visiting the echo state property. *Neural Netw.* **2012**, *35*, 1–9. [CrossRef]
43. Van Laarhoven, P.J.; Aarts, E.H. Simulated annealing. In *Simulated Annealing: Theory and Applications*; Springer: Berlin/Heidelberg, Germany, 1987; pp. 7–15. [CrossRef]
44. Lao, J.; Yan, M.; Tian, B.; Jiang, C.; Luo, C.; Xie, Z.; Zhu, Q.; Bao, Z.; Zhong, N.; Tang, X.; et al. Ultralow-Power Machine Vision with Self-Powered Sensor Reservoir. *Adv. Sci.* **2022**, *9*, 2106092. [CrossRef]
45. Lorenz, E.N. Deterministic nonperiodic flow. *J. Atmos. Sci.* **1963**, *20*, 130–141. [CrossRef]
46. Estrada, E.; Rodriguez-Velazquez, J.A. Subgraph centrality in complex networks. *Phys. Rev. E* **2005**, *71*, 056103. [CrossRef]
47. Bolland, J.M. Sorting out centrality: An analysis of the performance of four centrality models in real and simulated networks. *Soc. Netw.* **1988**, *10*, 233–253. [CrossRef]
48. Opsahl, T.; Agneessens, F.; Skvoretz, J. Node centrality in weighted networks: Generalizing degree and shortest paths. *Soc. Netw.* **2010**, *32*, 245–251. [CrossRef]
49. Barthelemy, M. Betweenness centrality in large complex networks. *Eur. Phys. J. B* **2004**, *38*, 163–168. [CrossRef]
50. Xing, W.; Ghorbani, A. Weighted pagerank algorithm. In Proceedings of the Second Annual Conference on Communication Networks and Services Research, Bhopal, India, 14–16 November 2004; IEEE: Piscataway, NJ, USA, 2004; pp. 305–314. [CrossRef]
51. Verstraeten, D.; Dambre, J.; Dutoit, X.; Schrauwen, B. Memory versus non-linearity in reservoirs. In Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, Spain, 18–23 July 2010; IEEE: Piscataway, NJ, USA, 2010; pp. 1–8. [CrossRef]
52. Berman, A.; Zhang, X.D. On the spectral radius of graphs with cut vertices. *J. Comb. Theory Ser. B* **2001**, *83*, 233–240. [CrossRef]
53. Ouyang, B.; Xia, Y.; Wang, C.; Ye, Q.; Yan, Z.; Tang, Q. Quantifying importance of edges in networks. *IEEE Trans. Circuits Syst. Express Briefs* **2018**, *65*, 1244–1248. [CrossRef]
54. Girvan, M.; Newman, M.E. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA* **2002**, *99*, 7821–7826. [CrossRef]
55. Bompard, E.; Wu, D.; Xue, F. Structural vulnerability of power systems: A topological approach. *Electr. Power Syst. Res.* **2011**, *81*, 1334–1340. [CrossRef]
56. Wu, T.; Wang, X.; Qiao, S.; Xian, X.; Liu, Y.; Zhang, L. Small perturbations are enough: Adversarial attacks on time series prediction. *Inf. Sci.* **2022**, *587*, 794–812. [CrossRef]

57. Zügner, D.; Akbarnejad, A.; Günnemann, S. Adversarial attacks on neural networks for graph data. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 2847–2856. [CrossRef]
58. Nirkin, Y.; Masi, I.; Tuan, A.T.; Hassner, T.; Medioni, G. On face segmentation, face swapping, and face perception. In Proceedings of the 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), Xi'an, China, 15–19 May 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 98–105. [CrossRef]
59. Hussain, S.; Neekhara, P.; Jere, M.; Koushanfar, F.; McAuley, J. Adversarial deepfakes: Evaluating vulnerability of deepfake detectors to adversarial examples. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Virtual, 5–9 January 2021; pp. 3348–3357. [CrossRef]
60. Li, L.; Bao, J.; Yang, H.; Chen, D.; Wen, F. Faceshifter: Towards high fidelity and occlusion aware face swapping. *arXiv* **2019**, arXiv:1912.13457.
61. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
62. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.