# Echo State Network with Adversarial Training

Takanori Akiyama[1](✉) and Gouhei Tanaka[1,2]

[1] Graduate School of Information Science and Technology,
The University of Tokyo, Tokyo 113-8656, Japan
{taka_akiyama,gouhei}@sat.t.u-tokyo.ac.jp
[2] Graduate School of Engineering,
The University of Tokyo, Tokyo 113-8656, Japan

**Abstract.** Reservoir Computing (RC) is a high-speed machine learning framework for temporal data processing. Especially, the Echo State Network (ESN), which is one of the RC models, has been successfully applied to many temporal tasks. However, its prediction ability depends heavily on hyperparameter values. In this work, we propose a new ESN training method inspired by Generative Adversarial Networks (GANs). Our method intends to minimize the difference between the distribution of teacher data and that of generated samples, and therefore we can generate samples that reflect the dynamics in the teacher data. We apply a feedforward neural network as a discriminator so that we don't need to use backpropagation through time in training. We justify the effectiveness of the proposed method in time series prediction tasks.

**Keywords:** Echo State Network · Recurrent Neural Network · Generative Adversarial Network · Nonlinear time series prediction

## 1 Introduction

Reservoir Computing (RC) has been widely researched as a fast machine learning method. RC is the Recurrent Neural Network (RNN) based framework that only trains the linear readout layer and fixes parameters in other layers before training. RC shows excellent performance in various benchmark tasks despite its simple training algorithm. Moreover, another major advantage is that RC is suitable for hardware implementation with a wide variety of physical systems [8].

Especially, Echo State Networks (ESNs) were initially proposed by Jaeger [2] and it was shown that they were useful in nonlinear time series prediction [4]. The underlying principle is that a well-trained ESN is able to reproduce the attractor of given nonlinear dynamical systems. However, the performance of the ESNs is significantly sensitive to the settings of hyperparameters such as the input scaling and the spectral radius of the reservoir connection weight matrix. In the case of physical implementation, it is hard to change and adjust such

I. V. Tetko et al. (Eds.): ICANN 2019, LNCS 11731, pp. 82–88, 2019.
https://doi.org/10.1007/978-3-030-30493-5_8

hyperparameters. Therefore, a reduction of the hyperparameter sensitivity only by changing a training method is regarded as an important research topic.

In this work, we incorporate the concept of Generative Adversarial Networks (GANs) [1] into ESNs to solve the abovementioned problem. A GAN consists of two networks, a discriminator to distinguish between teacher data and generated samples and a generator to deceive the discriminator. Original GANs can minimize the Jensen-Shannon divergence between the real data distribution and the generated one instead of minimizing the squared error. In our method, a generator is an ESN and a Deep Neural Network (DNN) based discriminator distinguishes between the real time series data and those generated by the ESN. Then we use the weighted sum of the conventional squared error and the adversarial loss. By introducing the adversarial loss, it is expected that the ESN can generate samples which reflect the dynamics underlying the given data better than the conventional ESN training based on the least squared error.

There are three major advantages in the proposed method. First, the prediction accuracy can be improved even when the settings of hyperparameters in the ESN are inappropriate. Only by introducing the adversarial loss in the training step, we can construct a high-quality predictor with 'bad' reservoirs. Second, the computational cost for training in our method is much smaller than that of RNNs. We use simple feedforward neural networks as a discriminator, instead of temporal neural networks like RNNs, to avoid using computationally expensive backpropagation through time (BPTT) [9] in training. Simultaneously, we introduce the concept of time-delay embeddings to construct the input to a discriminator. Therefore, we can consider time-dependent features in a discriminator without BPTT. Third, trained parameters in the ESN are only those in the readout layer, and therefore our method can be applied to other types of reservoir models and physical reservoirs.

We demonstrate the effectiveness of our method for benchmark nonlinear time series prediction tasks. Especially, when the settings of hyperparameters are not so appropriate, our training method outperforms the conventional one.

# 2 Methods

## 2.1 Echo State Network

The ESN uses an RNN-based reservoir composed of discrete-time artificial neurons. The time evolution of the reservoir state vector is described as follows:

$$\boldsymbol{r}(t + \Delta t) = \tanh\left[\boldsymbol{A}\boldsymbol{r}(t) + \boldsymbol{W}_{in}\boldsymbol{u}(t)\right], \ \hat{\boldsymbol{y}}(t) = \boldsymbol{W}_{out}\boldsymbol{r}(t), \tag{1}$$

where $\boldsymbol{r}(t)$ denotes the state vector of reservoir units, $\boldsymbol{W}_{in}$ is the weight matrix of the input layer, and $\boldsymbol{A}$ is the reservoir connection weight matrix. The readout layer gives the linear combination of the states of reservoir neurons with $\boldsymbol{W}_{out}$, which denotes the weight matrix of the readout layer.

In the training period $-T \leq t < 0$, the readout weight is adjusted so as to minimize the squared error between the predicted value $\hat{\boldsymbol{y}}(t)$ and the teacher output $\boldsymbol{y}(t)$ as follows:

$$\hat{\boldsymbol{W}}_{out} = \underset{\boldsymbol{W}_{out}}{\operatorname{argmin}} \sum_{-T\leq t<0} ||\boldsymbol{W}_{out}\boldsymbol{r}(t) - \boldsymbol{y}(t)||^2 + \beta ||\boldsymbol{W}_{out}||^2, \tag{2}$$

where $\beta > 0$ is the Tikhonov regularization parameter to avoid overfitting.

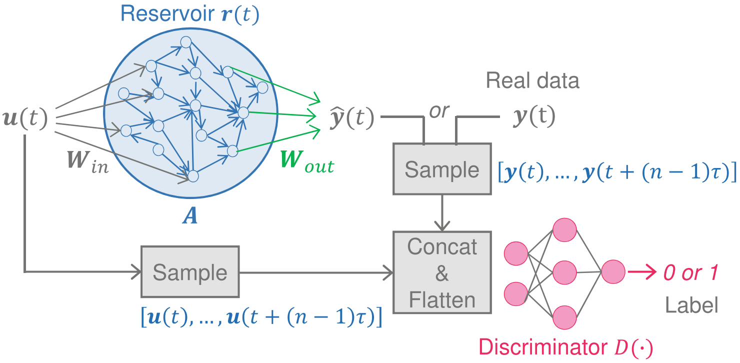

**Fig. 1.** Architecture of the ESN with adversarial training

### 2.2 Echo State Network with Adversarial Training

In our method, adversarial training [6] is used to optimize the readout weight in the ESN. Figure 1 shows the architecture of the proposed model. The discriminator $D$ is trained such that the output represents the probability that the input is drawn from teacher data. $\boldsymbol{u}(t)$ and $\hat{\boldsymbol{y}}(t)$ are embedded into the same time-delay coordinates, and then the concatenation of them are fed to the discriminator as

$$D\left(\begin{bmatrix} \boldsymbol{u}(t), \cdots, \boldsymbol{u}(t+(n-1)\tau) \\ \hat{\boldsymbol{y}}(t), \cdots, \hat{\boldsymbol{y}}(t+(n-1)\tau) \end{bmatrix}\right) := D(\boldsymbol{u}, \hat{\boldsymbol{y}}, t), \tag{3}$$

where $n$ and $\tau$ represent the dimension and time delay of the time-delay coordinates, respectively.

We define the discriminator loss as follows:

$$L_D = \mathbb{E}\left[-\log\left(1 - D(\boldsymbol{u}, \hat{\boldsymbol{y}}, t)\right)\right] + \mathbb{E}\left[-\log\left(D(\boldsymbol{u}, \boldsymbol{y}, \ \mathrm{t})\right)\right]. \tag{4}$$

Then we define the generator loss as follows:

$$L_G = w_D L_G^{ADV} + (1 - w_D)\frac{E_{L_{ADV}}}{E_{L_{SE}}} L_G^{SE}, \tag{5}$$

$$L_G^{ADV} = \mathbb{E}\left[-\log\left(D(\boldsymbol{u}, \hat{\boldsymbol{y}}, t)\right)\right], \ L_G^{SE} = \sum_{-T\leq t<0} ||\hat{\boldsymbol{y}}(t) - \boldsymbol{y}(t)||^2, \tag{6}$$

where $E_{L_{ADV}}$ and $E_{L_{SE}}$ denote the expectation values of $L_G^{ADV}$ and $L_G^{SE}$, respectively. $w_D$ is the weight of the adversarial loss in the generator loss. The discriminator loss $L_D$ and the generator loss $L_G$ are minimized alternately. Note that only $\boldsymbol{W}_{out}$ is optimized in the training of the generator. The procedure is formally presented in Algorithm 1.

# 3 Results

We demonstrate the effectiveness of the proposed method in prediction tasks with two benchmark time series, NARMA10 and the Lorenz system. In these two experiments, we set the reservoir size at 100 and the scarcity of the connection weight matrix at 0.95. Before adversarial training, we pretrained the output weight in the ESN using Tikhonov regularization with $\beta = 10^{-4}$. The architecture of the discriminator model is a feedforward network that consists of four hidden layers of 32 ReLU units and we set $n = 20$.

**Algorithm 1.** Stochastic gradient descent adversarial training of the ESN. We used $k_G = 2$ in our experiments.

```
Require: $n, \tau$: the parameter of time-delay embeddings, $w_D$: the weight of the adversarial loss, $m$: the batch size, $k_G$: the number of generator iterations per the discriminator iteration.
Pretrain the readout weight $\boldsymbol{W}_{out}$ using conventional least squares regression.
for number of training iterations do
    Sample a batch $t^{(1)}, \cdots, t^{(m)}$ from $-T \le t < 0$.
    Update the discriminator with the stochastic gradient method:
        $$\nabla_{\theta_D} \frac{1}{m} \sum_{i=1}^{m} \left[ -\log(1 - D(\boldsymbol{u}, \hat{\boldsymbol{y}}, t^{(i)})) - \log D(\boldsymbol{u}, \boldsymbol{y}, t^{(i)}) \right]. \quad (7)$$
    Calculate $E_{L_{ADV}}$ and $E_{L_{SE}}$.
    for $k_G$ steps do
        Sample a batch $t^{(1)}, \cdots, t^{(m)}$ from $-T \le t < 0$.
        Update $\boldsymbol{W}_{out}$ with the stochastic gradient descent method:
            $$\nabla_{\boldsymbol{W}_{out}} \left( \frac{\omega_D}{m} \sum_{i=1}^{m} \left[ -\log D(\boldsymbol{u}, \hat{\boldsymbol{y}}, t^{(i)}) \right] + (1 - \omega_D) \frac{E_{L_{ADV}}}{E_{L_{SE}}} L_G^{SE} \right). \quad (8)$$
    end for
end for
```

## 3.1 NARMA10

The NARMA10 task [3] is the identification of the order-10 discrete-time nonlinear dynamical system. We used $-900 \le t < 0$ for training and $0 \le t < 100$ for testing. In this task, we set the input scaling of the ESN at 1.0. Time delay in embeddings is $\tau = 1$. We conducted experiments for two different cases, where the spectral radius is good ($\rho = 0.8$) and bad ($\rho = 0.4$).

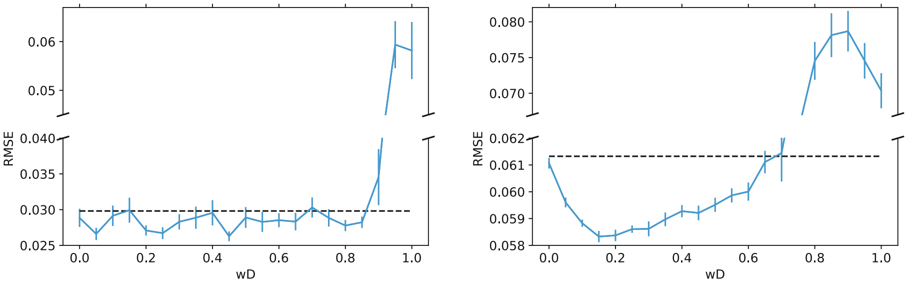

**Fig. 2.** RMSEs for different values of $w_D$ in the NARMA10 task. The spectral radius of $A$ is set at 0.8 (left) and 0.4 (right). Block dotted lines are RMSEs in the ESN with the least squared method. The error bars represent normalized errors.

Figure 2 shows the root mean squared error (RMSE) for the NARMA10 task, plotted against the value of $w_D$. In the case with $\rho = 0.8$, we can see that the prediction performance is improved by the introduction of the adversarial loss in some settings. In addition, in the case with $\rho = 0.4$, the prediction accuracy for $0.05 \leq w_D \leq 0.6$ is lower than the case when we use only the squared error. From this result, we can conclude that the adversarial loss in the ESN training improves the prediction accuracy, especially when the settings of the hyperparameter in the ESN are not so good.

### 3.2 Lorenz Systems

The Lorenz system [5] is a continuous-time nonlinear dynamical system which shows chaotic bahavior and is described by the following differential equations:

$$\begin{aligned}\frac{dx}{dt} &= 10(y - x),\\ \frac{dy}{dt} &= x(28 - y) - y,\\ \frac{dz}{dt} &= xy - \frac{8}{3}z.\end{aligned} \tag{9}$$

In this experiment, we predict the first variable $x(t)$ to evaluate the performance of the proposed model. We used $-100 \leq t < 0$ for training and $0 \leq t < 25$ for testing and set $\Delta t = 0.02$. The input scaling of the ESN is 0.1 and time delay $\tau$ is 0.08. We conducted an experiment for a bad parameter setting where the spectral radius $\rho = 0.4$.

Figure 3 shows the RMSE in this task, plotted against the value of $w_D$. The proposed method improves the prediction performance compared with the ESN

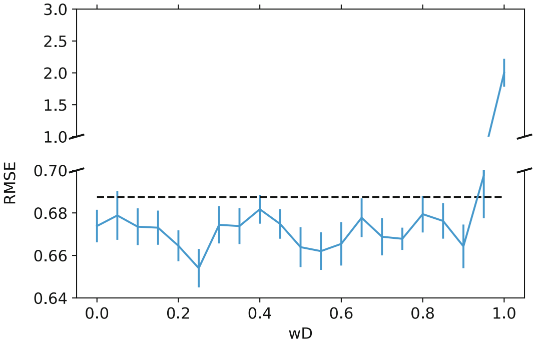

**Fig. 3.** RMSEs for different values of $w_D$ in the prediction of the first variable of the Lorenz system. Block dotted lines are RMSEs in the ESN with the least squared method. The error bars represent normalized errors.

with the conventional training in a wide range of $w_D$ (the optimal setting of $w_D$ appears 0.25). Our proposed method uses the concept of time-delay embeddings, and thus we can conclude that the generator can reflect the overall dynamics even when we can observe only one variable on the basis of Takens Embedding Theorem [7].

## 4 Conclusion

In this work, we proposed a new ESN training method using adversarial training where the loss function is described as the weighted sum of the conventional squared error and the adversarial loss. Then we demonstrated that the proposed method can improve the prediction accuracy in nonlinear time series prediction tasks. In future work, we will test another model as a discriminator and check the effectiveness for other tasks.

**Acknowledgment.** This work was partially supported by JSPS KAKENHI Grant Number JP16K00326 (GT) and based on results obtained from a project subsidized by the New Energy and Industrial Technology Development Organization (NEDO).

## References

1. Goodfellow, I., et al.: Generative adversarial nets. Adv. Neural Inf. Process. Syst. **27**, 2672–2680 (2014)
2. Jaeger, H.: The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, GMD - German National Research Institute for Computer Science (2001)
3. Jaeger, H.: Adaptive nonlinear system identification with echo state networks. In: Proceedings of the 15th International Conference on Neural Information Processing Systems, pp. 609–616 (2002)

4. Jaeger, H., Haas, H.: Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. Science **304**(5667), 78–80 (2004)
5. Lorenz, E.N.: Deterministic nonperiodic flow. J. Atmos. Sci. **20**(2), 130–141 (1963)
6. Saito, Y., Takamichi, S., Saruwatari, H.: Statistical parametric speech synthesis incorporating generative adversarial networks. IEEE/ACM Trans. Audio Speech Lang. Process. **26**(1), 84–96 (2018)
7. Takens, F.: Detecting strange attractors in turbulence. In: Rand, D., Young, L.-S. (eds.) Dynamical Systems and Turbulence, Warwick 1980. LNM, vol. 898, pp. 366–381. Springer, Heidelberg (1981). https://doi.org/10.1007/BFb0091924
8. Tanaka, G., et al.: Recent advances in physical reservoir computing: a review. Neural Netw. **115**, 100–123 (2019)
9. Werbos, P.J.: Backpropagation through time: what it does and how to do it. Proc. IEEE **78**(10), 1550–1560 (1990)