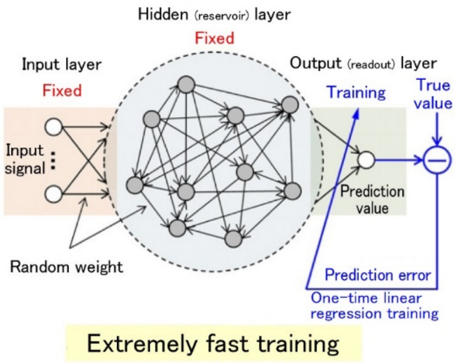
"Reservoir Computing" – Time-series Prediction Method for Fast and Accurate Soft Sensor Modeling



SHUNSUKE TAKAGAKI[*1] KOKI TATEISHI[*2]
HIROYASU ANDO[*3]

In chemical plants and industrial plants, "soft sensors" is known as a technology to estimate state quantities that cannot be measured directly. Their application for purposes such as control and monitoring are progressing. Recent technological developments in deep learning are remarkable. While their employment in soft sensors has enabled highly accurate estimation, the problem lies in the necessity of a longer training time. To address this problem, we built a soft sensor using "Reservoir Computing," with which training can be done in an extremely short time while retaining the high accuracy of estimation. This report presents an overview of Reservoir Computing and its application using a case study on the prediction of calcium carbonate concentration in flue gas desulfurization equipment.

1. Introduction

In recent years, AI technologies such as machine learning and deep learning have been in rapid development. Their application in addressing industrial problems, particularly time-series prediction of unknown state quantities, is being actively promoted. For example, some types of state quantities in the field of plants are important yet difficult to measure. If models to predict such state quantities in real time can be built, it can be expected that these will be used as an alternative to hardware sensors, reducing sensor operational costs and improving plant operability through the application to anomaly detection and control. Such prediction models, which can replace hardware sensors, are generally called "soft sensors". Highly accurate modeling is possible by allowing AI-based time-series prediction models to be trained on the operation data measured/collected in the past.

A Recurrent Neural Network (RNN), which is a kind of deep learning model, is one of the well-known representative methods of AI-based time-series prediction. However, RNNs generally are trained by repeatedly adjusting a large number of internal parameters of models, which are so numerous that countless hours of training are required if models are trained on long-term operation data. For this reason, there were two bottlenecks in the practical application of AI-based highly accurate soft sensors. The first was the inability to make full use of collected data for training, because of the limited computing resources (e.g., using edge computers installed at a plant) making it necessary, for example, to undersample operational datasets or limit training data. The second was the inability to readily update soft sensors according to temporal changes.

"Reservoir Computing" is referred to as a possible solution to these problems in the context of time-series prediction technology (detailed in Chapter 2). One of its well-known representative methods is an "Echo State Network (ESN)[(1)]", which is characterized by extremely short training time and no need for repeated adjustment of internal parameters despite it being an RNN capable of forecasting the time series dynamics with high accuracy. Mitsubishi Heavy Industries, Ltd. has arranged the ESN design guidelines under the academic guidance of Tohoku University. The effectiveness of ESN in the creation of soft sensors has been verified in terms of both training time

*1 CIS Department, Digital Innovation Headquarters, Mitsubishi Heavy Industries, Ltd.
*2 Chief Staff Manager, CIS Department, Digital Innovation Headquarters, Mitsubishi Heavy Industries, Ltd.
*3 Professor, Advanced Institute for Materials Research, Tohoku University

and prediction accuracy by examining its application using operation data from actual plants. The summary is given in this report.

2. Reservoir Computing technology: ESN

Reservoir Computing is a modeling technique for time series based on the concept of (a) mapping input time series into a high-dimensional space and (b) reading desired information from the mapped space. For example, when we throw a couple of pebbles into a reservoir, complex ripples are formed on the water. Let us suppose that these ripples contain information about the pebbles (such as mass, shape and trajectory), then it would be somehow possible to make approximations in this regard from the ripples. The term "Reservoir Computing" came from this metaphor. It can be said that the sequence of physical phenomena, "starting from the pebbles' motion to the formation of ripples", corresponds to (a) of the above, while "estimation of relevant information based on the formed ripples pertains" corresponds to (b) of reading information capable[2]. Reservoir Computing focuses on (a) mapping into a given high-dimensional space and (b) figuring out how to read desired information from it. Described below is the ESN, which is an RNN based on this concept and is one of the paradigms of Reservoir Computing.

Given in **Figure 1** are the conceptual diagrams of typical RNN and ESN architectures and training methods. Consisting of three layers (input, hidden and output) each with a network of connected neurons, both RNNs and ESNs train by adjusting the strength of neural connections (called "weight") to produce a desired input-output relationship. However, their training methods are quite different from each other. In typical RNN training, time series are inputted in sequence before prediction values are computed, which is followed by simultaneous adjustment of all weights of input, hidden and output layers according to the assessment of prediction error. This series of processes are repeated, until the prediction error becomes small enough. In ESNs, on the other hand, the weights of both input and hidden layers are fixed after random initialization. Training of the weights of the output layer is done considering the output layer as a simple linear combination layer. When the same input is given, the output of the hidden layer (which is the input to the output layer) is invariant if the input and hidden layers have fixed weights. Performing linear regression once therefore suffices to train the weights of the output layer, making repeated weight adjustment unnecessary. Thus, ESNs are most characterized by their extremely fast training capability while time-series modeling as complex as RNNs is possible. In ESNs, furthermore, the hidden layer and the output layer correspond to (a) of the above and (b), respectively. They are also called the reservoir layer and the readout layer, respectively.

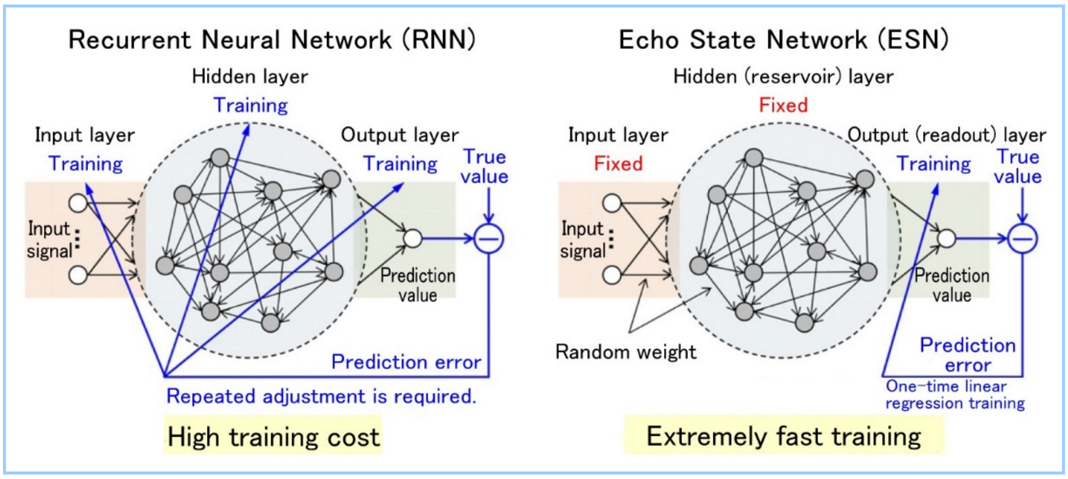


Figure 1 Conceptual comparison between typical RNNs and ESNs[2]

3. ESN design guidelines

Figure 2 is a block diagram showing a typical ESN architecture. The input data are given as vectors, which are linearly transformed by the weight matrix $\boldsymbol{W}_{\mathrm{in}}$ in the input layer. In the reservoir layer, the linearly transformed data are further transformed through the activation function $\boldsymbol{f}$ into reservoir state vectors. The use of previous reservoir states in this transformation process enables time-series modeling. In the readout layer, the reservoir state vectors are combined with the input data, which are then linearly transformed by the weight matrix $\boldsymbol{W}_{\mathrm{out}}$ to produce the final output of prediction values. ESNs are a simple method in which only this $\boldsymbol{W}_{\mathrm{out}}$ is trained. For this reason, designing an appropriate model architecture in advance is the key to achieving an accurate input-output relationship. This chapter pertains to ESN design guidelines for the practical application to soft sensors.

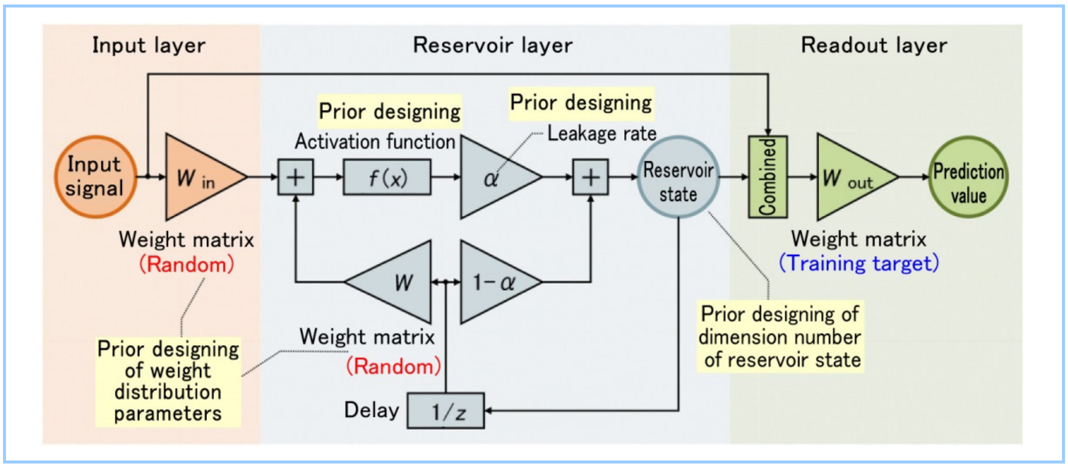


Figure 2　Typical ESN block diagram

3.1　Automatic adjustment of hyperparameters

The ESN architecture shown in Figure 2 (except for the part of the readout layer) is mainly determined by the dimension of the reservoir state, the parameters of random weight distributions for the input and reservoir layers, and parameter α (called the leakage rate) that adjusts the speed of the dynamics represented. These parameters are called hyperparameters. While the optimality of hyperparameters depends on a given prediction task, some empirical guidelines for adjustment are known(3), but a definite methodology has yet to be established. Manual adjustment has therefore been needed so far. The previous study reported the effectiveness of Bayesian optimization for hyperparameter adjustment(4). Such technology has enabled labor-saving automatic adjustment.

3.2　Selection of appropriate activation function

A non-linear function needs to be applied as the activation function, if the approximation capability of ESN is to be enhanced. The tanh function is used in most cases. While this tanh function is odd, it is known that the use of an even function such as a sinc function ($y = \sin(x)/x$) can improve the ESN accuracy for certain prediction tasks(5). Like in the case of hyperparameters, addressed in the preceding section, the optimality of activation function depends on a given prediction task. This requires other activation functions than the tanh to be tested for appropriate selection.

3.3　Multi-step-ahead prediction and linear interpolation

In typical ESNs, the prediction value is outputted one-step ahead for each of the evenly spaced inputs. However, when the timestep is small relative to the dynamics of the prediction target (i.e., when a slow response system is modeled using small time interval data), the amount of change made in every timestep becomes relatively small. If this is the case, there is a concern that such models may become susceptible to noise when they are trained on actual data. A measure against this problem is considered to train ESNs to predict multi-steps ahead, so that these ESNs can predict one-step ahead by performing linear interpolation between the predicted value at a given time and the value obtained by multi-step-ahead prediction.

3.4　Ensemble of models

The method called “ensemble” involves preparing multiple prediction models and providing the average (or median, etc.) of their prediction values as the output of the models as a whole. It is a technique generally used in the fields of machine learning and deep learning to improve the accuracy and stability of prediction. Because of their principle that the weights of input and reservoir layers are fixed on a random basis, ESNs intrinsically hold the risk of destabilizing the prediction. Although complete avoidance is impossible, it is possible to reduce the risk to some extent with the use of ensemble.

The conceptual diagrams of the aforementioned four points of design guidelines are shown in **Figure 3**.

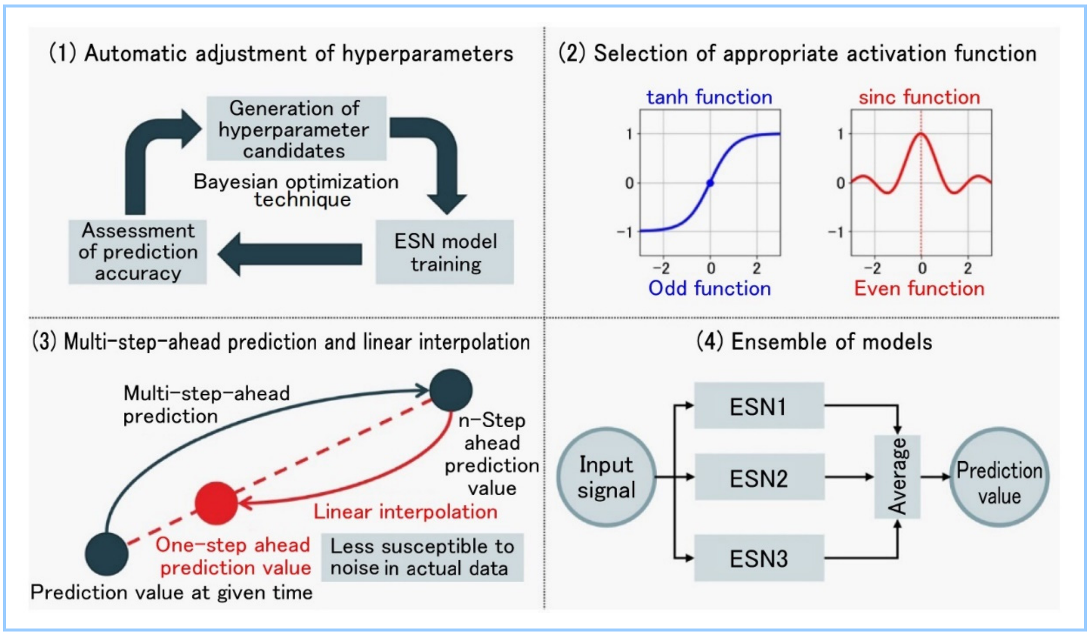


Figure 3 ESN design guidelines for creating soft sensors

4. Verification using data from actual plant

This chapter describes an example of the application of ESN to the prediction of calcium carbonate concentration in flue gas desulfurization equipment in order to verify the effectiveness of soft sensor building by ESN. Conventionally, regarding the flue gas desulfurization equipment of our company, we have provided an analyzer to customers to measure the calcium carbonate concentration in the absorption tower for operation monitoring and control purposes. Problems such as blockage in the sampling tube necessitate periodic maintenance. Under such circumstances, we have been working on the development of soft sensors for estimating the calcium carbonate concentration, as part of our undertakings to realize AI-based optimal facility operation/maintenance and develop digital solutions for reduced operational costs. Highly accurate soft sensors using deep learning have already been realized[(6)]. However, as high training costs remained problematic, we attempted to use ESNs for enabling fast and accurate training of soft sensors.

In this attempt, about two months' worth of operation data from existing flue gas desulfurization equipment were used to train ESN models to predict the calcium carbonate concentration based on eight explanatory variables such as SO_2 concentrations at the inlet and outlet of the absorption tower. For this purpose, we followed the ESN design guidelines described in the previous chapter, that is, using a sinc function as the activation function (Section 3.2), performing 15-step-ahead prediction (Section 3.3), allowing 10 models to be trained while automatically adjusting the hyperparameters (Section 3.1), and building ensembles of these 10 models (Section 3.4).

Figure 4 shows the cross-validated results of the accuracy of calcium carbonate concentration prediction by this ESN, together with the model's training time per sample. These results are compared with those of a Long Short-Term Memory (LSTM), which is a well-known RNN[(7)]. While the ESN prediction accuracy is as high as or higher than that of LSTM, the training time is about 1/140 of LSTM. This has demonstrated that fast and accurate training of soft sensors is possible with ESNs.

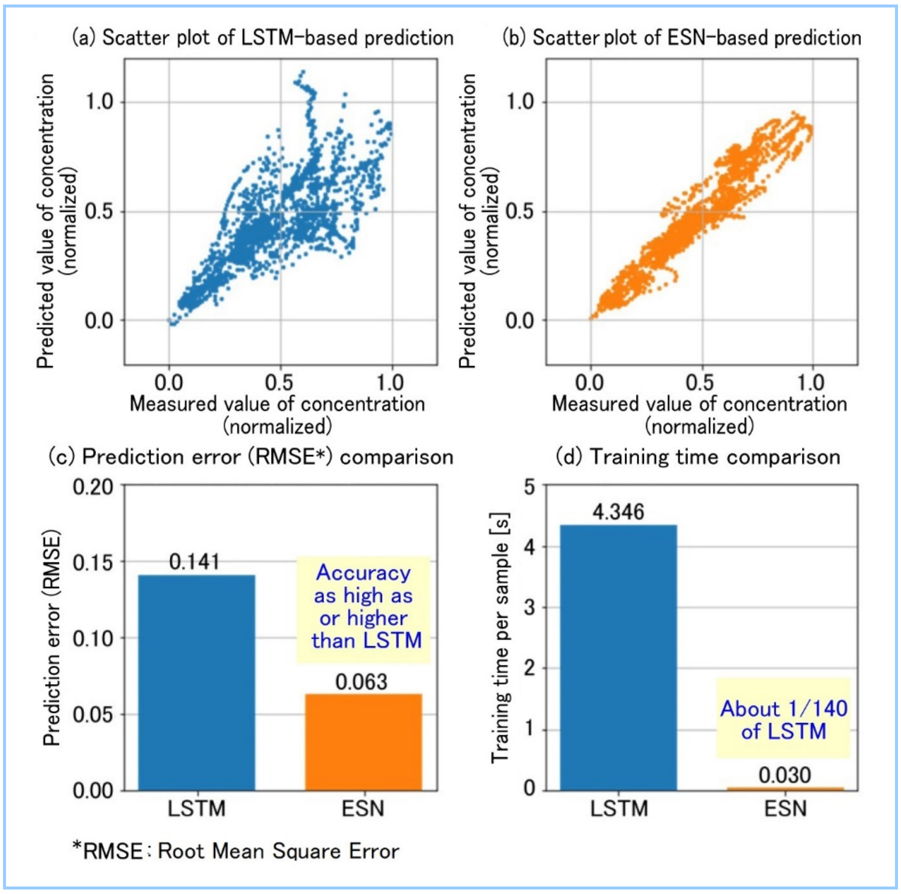


Figure 4 Prediction accuracies and model training times of ESN and LSTM for prediction problem of calcium carbonate concentration

5. Conclusion

ESNs are a well-known method of Reservoir Computing, which is a technology for enabling fast and accurate time-series prediction. This report summarizes ESN design guidelines for the practical application to soft sensors. Following the design guidelines, we attempted to train ESN models for soft sensors to compute the calcium carbonate concentration in flue gas desulfurization equipment. The results indicate that the training time can be reduced to about 1/140, despite the accuracy of soft sensors being as good as or better than the conventional deep learning method. However, ESN-based soft sensors still face some challenges, including (1) further reduction of the risk of prediction instability resulting from weight randomness, and (2) establishment of training method under the conditions in which it is impossible to continuously obtain training data because of reasons such as removal of permanent devices. We will develop technologies that can solve these problems as well as promote the expansion of application range in our products.

References

(1) Jaegar, H., The “echo state” approach to analysing and training recurrent neural networks, German National Research Center for Information Technology Report Vol.148 (2001)
(2) Gouhei Tanaka, Concept of Reservoir Computing and Its Recent Trends, The Journal of The Institute of Electronics, Information and Communication Engineers Vol.102 No.2 (2019) p.108-113
(3) Lukoševicius, M., A practical guide to applying Echo State Networks, Neural Networks: Tricks of the Trade, Springer, (2012) p.659-686
(4) Racca, A. et al., Robust optimization and validation of echo state networks for learning chaotic dynamics, Neural Networks Vol.142 (2021) p.252-268
(5) Chang, H. et al., Effect of shapes of activation functions on predictability in the echo state network, arXiv preprint arXiv:1905.09419 (2019)
(6) Yukiko Agata et al., Intelligent Solution TOMONI® for Advanced Maintenance and Operation of Critical Infrastructure, Mitsubishi Heavy Industries Technical Review Vol.59 No.3 (2022)
(7) Hochreiter, S. et al., Long Short-Term Memory, Neural Computation Vol.9 No.8 (1997) p.1735-178