*mathematics*

*Article*

# Event-Triggered Relearning Modeling Method for Stochastic System with Non-Stationary Variable Operating Conditions

**Jiyan Liu [1,2], Yong Zhang [1,2,*], Yuyang Zhou [3,*] and Jing Chen [4]**

[1] School of Automation and Electrical Engineering, Inner Mongolia University of Science and Technology, Baotou 014010, China; 2021023289@stu.imust.edu.cn
[2] Key Laboratory of Synthetical Automation for Process Industries at Universities of Inner Mongolia Autonomous Region, Inner Mongolia University of Science and Technology, Baotou 014010, China
[3] School of Computing Engineering and Built Environment, Edinburgh Napier University, Edinburgh EH10 5DT, UK
[4] School of Science, Jiangnan University, Wuxi 214122, China; 8201703038@jiangnan.edu.cn
[*] Correspondence: zhangyong@imust.edu.cn (Y.Z.); y.zhou@napier.ac.uk (Y.Z.)

**Abstract:** This study presents a novel event-triggered relearning framework for neural network modeling, designed to improve prediction precision in dynamic stochastic complex industrial systems under non-stationary and variable conditions. Firstly, a sliding window algorithm combined with entropy is applied to divide the input and output datasets across different operational conditions, establishing clear data boundaries. Following this, the prediction errors derived from the neural network under different operational states are harnessed to define a set of event-triggered relearning criteria. Once these conditions are triggered, the relevant dataset is used to recalibrate the model to the specific operational condition and predict the data under this operating condition. When the predicted data fall within the training input range of a pre-trained model, we switch to that model for immediate prediction. Compared with the conventional BP neural network model and random vector functional-link network, the proposed model can produce a better estimation accuracy and reduce computation costs. Finally, the effectiveness of our proposed method is validated through numerical simulation tests using nonlinear Hammerstein models with Gaussian noise, reflecting complex stochastic industrial processes.

**Keywords:** stochastic processes; non-stationary and variable conditions; event-triggered conditions; sliding window algorithm; information entropy

**MSC:** 68T07; 93-10

**Citation:** Liu, J.; Zhang, Y.; Zhou, Y.; Chen, J. Event-Triggered Relearning Modeling Method for Stochastic System with Non-Stationary Variable Operating Conditions. *Mathematics* **2024**, *12*, 667. https://doi.org/10.3390/math12050667

Academic Editor: Leonid Piterbarg

Received: 24 January 2024
Revised: 18 February 2024
Accepted: 21 February 2024
Published: 24 February 2024

## 1. Introduction

Stochastic processes are ubiquitous in both natural and engineered systems, where they often exhibit complex and unpredictable behaviors that challenge conventional modeling and control methods [1–3]. In industrial contexts, stochastic systems may operate under multiple conditions due to mechanical breakdowns, environmental disturbances, or operator mistakes. These systems can unpredictably transition between different states, causing operational inefficiencies, potential harm, safety risks, and failures in control systems. Precise modeling of such systems is crucial for designing effective controllers and ensuring robust operation. Traditional approaches often employ neural networks to model complex industrial stochastic processes characterized by dynamic behaviors, unobservable internal states, and limited data confined to inputs and outputs. However, these conventional neural network models may fall short in accurately predicting the behavior of stochastic systems with multiple operating conditions. Taking the ironmaking blast furnace as an example, under stable operating conditions, the model can be effectively constructed using an offline training dataset to deliver accurate predictions [4–7]. However, ironmaking blast furnaces are often subject to random perturbations from the external environment, uncertainties within the system, and measurement errors, all of which have impacts on the operating condition and product quality of the blast furnace, manifesting as mismatches between the forecast data's boundary range and the boundary range of the modeling training input data, along with irregular changes. In this case, the prediction accuracy of the original neural network model will be significantly reduced.

In recent years, given the notable drop in prediction accuracy of conventional neural networks under non-stationary variable working conditions inherent in complex industrial stochastic processes, researchers and scholars have pursued research into online learning modeling algorithms to address these challenges. Zhou et al. [8] combined integrated auto-encoder and principal component analysis (PCA) techniques to propose a new type of improved random vector functional-link networks (RVFLNs) algorithms (AE-P-RVFLNs algorithms) for online estimation. Su et al. [9] established a multi-layer extreme learning machine (ML-ELM) predictive model for unsupervised sequential learning layer-by-layer. Zhou et al. [10] applied the output self-feedback structure to the traditional online sequential random vector functional-link networks (OS-RVFLNs), and proposed a dynamic modeling method. Jiang et al. [11] proposed a silicon content prediction method for adaptive working condition clustering of process variables, establishing Elman neural network submodel. He et al. [12] proposed to integrate bagging strategies, point-of-care learning methods, and semi-supervised extreme learning machines into a unified local semi-supervised model (BLSM) for online measurement. However, in the actual production process, due to the influence of interference such as detection instruments and transmitter failure or human operation, the data distribution varies greatly. Continuous updates and parameter adjustments of the model need to rely on previous experience, and then, the model cannot adapt well to the new data distribution, resulting in a decrease in model accuracy. Furthermore, there is no necessity for real-time updating of model parameters, which would otherwise escalate computational overhead. This consideration becomes particularly critical when processing large-scale datasets or managing high-dimensional data, where computational demands can become exceedingly burdensome.

An event-triggered mechanism is an automated event-based response mechanism [13,14]. Through an event-triggered mechanism, predefined rules or conditions activate the system to autonomously perform the appropriate actions or tasks upon their fulfillment. This study presents the design of an event-triggered controller gain for a larger sampling interval in [15]. In sewage treatment plants, the control strategy based on online optimization usually faces high computational complexity, so an event-triggered method solves the above problems in [16]. The backstepping technique is employed to design a controller by combining event-triggered mechanisms (ETMs), which can alleviate data transmission and save communication resource in [17]. A Zeno-free event-triggered mechanism in controller-to-actuator channels is proposed to save limited communication resources in [18]. A dynamic event-triggered control scheme for strict-feedback uncertain nonlinear systems is proposed, and a control scheme can ensure zero tracking stabilization error in the absence of Zeno behavior in [19].

Inspired by the aforementioned problems, the concept of an event-triggered mechanism is employed in this paper to counteract the decrease in neural network prediction accuracy due to non-stationary changing working conditions in complex industrial stochastic processes. This mechanism activates appropriate control actions in response to system state variations, facilitating adaptive control of the system. Firstly, both information entropy and sliding windows can be used in data processing and analysis. By combining them, the input data are detected to find the locations of the data jump point and to count the boundary range to divide the input and output datasets corresponding to different boundaries. Secondly, the prediction errors of the current model for data with different boundary ranges are calculated, and then, the event-triggered condition group is constructed from the above errors. When the boundary range of the prediction data diverges from that of the training input data of the current model and meets the specified triggered conditions, the model's parameters are updated by retraining with the dataset corresponding to the new boundary range. Conversely, if the boundary range of the prediction data aligns with the training input data of the current model, we simply transition to the model specific to that boundary range to forecast new data.

In conclusion, by integrating an event-triggered mechanism with a neural network model, we can effectively model and forecast the behavior of stochastic systems. According to the algorithm combining sliding window and entropy, we can detect and divide the boundary range of the input data, so as to distinguish different working conditions, and finally improve the prediction accuracy of the model under different working conditions. The contribution of this paper can be summarised as follows:

1 By integrating the sliding window technique with entropy, our approach successfully detects and segments the boundary range of input data, thereby efficiently categorizing various working conditions.

2 The prediction accuracy of the model decreases due to the change in working conditions. To address this, prediction errors are utilized to establish a group of event-triggered conditions, which guide the model's retraining process across diverse working conditions.

3 Distinct predictive models are developed corresponding to the segregated boundary ranges of input data. When the prediction data boundaries align with those of a pre-existing model's input, the system seamlessly transitions to the appropriate model for prediction.

The remainder of this paper is organized as follows. Section 2 describes the problem that conventional neural networks approximate the stochastic systems under the non-stationary and variable conditions. Section 3 develops the algorithm combining sliding window and information entropy and is applied to a modeling strategy for event-triggered relearning. Section 4 provides an example to show the effectiveness of the proposed method. Finally, we give some concluding remarks in Section 5.

## 2. Problem Description

### 2.1. System Statement

In practical applications, a wide array of engineering and control systems, ranging from aircraft and robots to power systems, exhibit stochastic characteristics [20–22]. A description of a stochastic system can be expressed as follows:

$$Y(k+1) = f(U(k), Y(k)) + V(k), \tag{1}$$

where $Y(k+1)$ is the output of the system at the discrete time $k+1$, and $U(k)$ and $Y(k)$ are the input and output of the system at the discrete time $k$, respectively. $V(k)$ denotes stochastic noise. $f(\cdot)$ represents nonlinear transformations. Compared with linear systems, stochastic nonlinear systems usually exhibit more complex and unpredictable dynamic behaviors. Therefore, the modeling and prediction of stochastic systems become crucial and essential areas of research. There are many methods for modeling and prediction. In this paper, we focus on applying neural network algorithms for this purpose. Detailed explanations and methodologies will be provided in the following section.

### 2.2. Neural Networks Modeling Problem

Neural networks are a common data-driven modeling method that applies input and output datasets to construct a model and predict unknown data for complex industrial stochastic processes [23,24]. For the stochastic system described above, the inputs and outputs are utilized as the respective input and output data for a neural network. Through data acquisition, we gather '$L$' sets of scalar data points, comprising '$m$' input variables and '$n$' output variables. This can be formally presented as follows:

$$U = [u_1, u_2, \cdots, u_m]^T \in R^{m\times 1}, \tag{2}$$

$$Y = [y_1, y_2, \cdots, y_n]^T \in R^{n\times 1}, \tag{3}$$

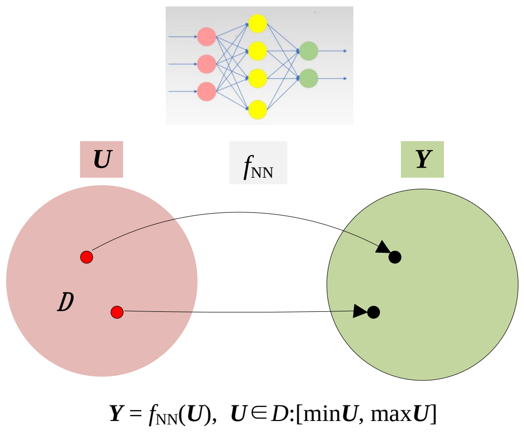
where $u_i$ is the time series of the $i$th input, and $y_j$ is the time series of the $j$th output. The time series of input and output can be expressed as follows:

$$\{U(k), Y(k)|k = 1, 2, \cdots, L\}. \tag{4}$$

Under stationary conditions, conventional neural networks use datasets to establish a nonlinear mapping relationship from input to output. The default condition is that values of the prediction data vary within the boundaries of its modeling training input data, which can be given as follows in Equation (5):

$$Y = f_{NN}(U), U \in D. \tag{5}$$

Just as the value of the independent variable in a function is constrained by a defined domain, as shown in Figure 1, data within the boundary range $D$ can be accurately predicted.



U
$f_{\mathrm{NN}}$
Y
D
$\boldsymbol{Y} = f_{\mathrm{NN}}(\boldsymbol{U})$, $\boldsymbol{U} \in D$:[min$\boldsymbol{U}$, max$\boldsymbol{U}$]

**Figure 1.** The prediction data is constrained by the boundary range of the model training data. The red dots represent different data points. The pink circle represents the boundary range of the input data, similar to the defined domain of a function. The green circle represents the boundary range of values predicted by the neural network for the input data, similar to the range of values of a function.

However, in the actual production process affected by the operating environment and human factors, the system working conditions will change, that is, cause changes in the system. During the nonlinear system, $f_j(\cdot), j = 1, 2, \cdots$, represent different nonlinear transformations. Simultaneously, the range of the system's inputs also shifts due to this change, leading to a corresponding variation in the output, which can be demonstrated in Equation (6) as follows:

$$Y(k+1) = \begin{cases} f_1(U(k), Y(k)), U(k) \in D_1 \\ f_2(U(k), Y(k)), U(k) \in D_2 \\ \quad \vdots \\ f_j(U(k), Y(k)), U(k) \in D_j \end{cases}, \tag{6}$$

where $D_i, i = 1, 2, \cdots, j$, represents the boundary range of the input variable.

In view of the above situation, conventional neural networks cannot accurately predict the prediction data of different boundary ranges. It is not possible to construct a one-to-one correspondence between different domains and different mappings similar to the segmentation function. Taking the multiple-input and multiple-output of nonlinear model with Gaussian noise as the research object to simulate complex industrial stochastic processes, the problem of model prediction accuracy degradation caused by neural network under non-stationary variable working conditions can be regarded as the problem of the decline in prediction accuracy of the original model caused by the inconsistency from the boundary range between the prediction data and the modeling training input data:

1 When the boundary range of the prediction data exceeds the boundary range of the training input data of the current model, how to detect the different boundaries of prediction data and redivide the input and output datasets.
2 When the boundary range of the prediction data is detected, the data for the variable working conditions prediction accuracy of the established model will decrease, how to improve the prediction accuracy of the model.
3 As time goes by, when the boundary range between the prediction data and training input data of established model will be consistent again, how to achieve the prediction of the data when the boundary range is the same.

In order to solve the above problems of conventional neural network modeling, the following relearning modeling strategy based on event-triggered mechanism is proposed.

## 3. Triggered Relearning Modeling Method

### 3.1. Modeling Strategy

The conventional neural network modeling strategy uses the input and output data under the stationary working condition, iteratively optimized through the backpropagation of error based on the gradient descent method [25,26]. The output weight is ultimately determined in line with the regularization principle to form a predictive model. A diagram of this conventional modeling strategy is depicted in Figure 2. When the prediction data satisfies the constraints of the boundary range of its modeling training input data, it can make accurate prediction with the data. However, when the range of the prediction data exceeds the boundary range of the modeling training input data, the conventional neural network will not be able to accurately learn the features of the data outside the range, resulting in inaccurate data prediction. To solve the aforementioned problems, a neural network modeling strategy based on triggered relearning is proposed, as shown in Figure 3.

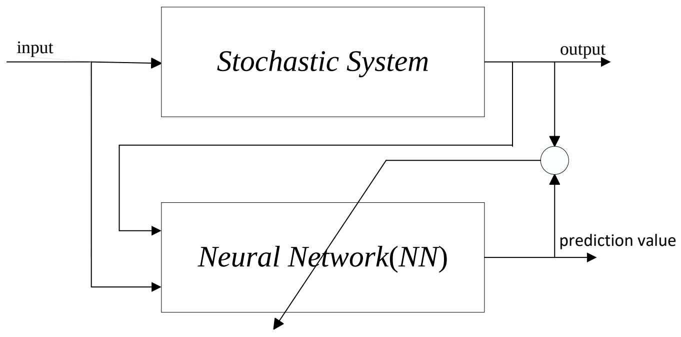


**Figure 2.** General modeling strategy.

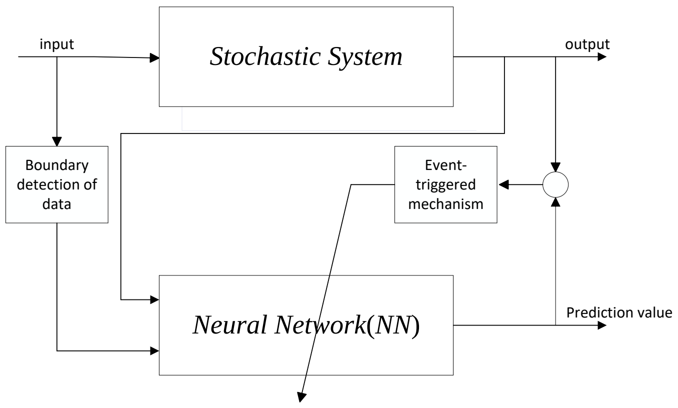


**Figure 3.** Event-triggered modeling strategy.

The process of triggered relearning modeling mainly solves the problems mentioned earlier in Section 2, and the specific solutions are given as follows:

1 To address the issue of discrepancies of the boundary ranges between the prediction data and the training input data of the current model, a method that combines sliding window techniques with information entropy is employed. This approach is mainly applied to detect and define the boundaries of the prediction data, resulting in a reclassification of the input and output datasets accordingly.

2 As we mentioned earlier, to address that issue of decline for the model's prediction accuracy, a modeling strategy for event-triggered relearning is proposed. This strategy is implemented to construct an event-triggered condition group, setting reasonable thresholds. If the triggered conditions are met, the model will be re-trained based on the datasets corresponding to the boundaries of the prediction input data.

3 If the same boundary range of prediction data recurs, as the boundary range has already been identified, segmented, and an associated prediction model established, the process will simply involve switching directly to the relevant model based on the current data's boundary range for subsequent predictions.

## *3.2. Dataset Boundary Detection Algorithm Based on Sliding Window and Information Entropy*

If the system operates under non-stationary conditions, it will cause some parameter transition points. A jump dataset can be understood as a collection of data that is affected by the system and has abrupt and drastic changes in the data flow. When a jump dataset occurs, it usually leads to a change in the distribution of the data, causing bias and discontinuities in the data. Therefore, when performing data-driven modeling, the jump dataset needs to be identified and processed to ensure the accuracy and reliability of the model.

Focused on the above challenges, this paper proposes a sliding window and information entropy method for detecting datasets that contain abrupt changes. Sliding window [27,28] analysis can reveal hidden patterns and information in time series data, providing a solid foundation for data analysis and modeling. Information entropy, as defined by Shannon, is a metric for quantifying the randomness of a discrete random variable. It is commonly used to measure the degree of signal variation and information content, and can be applied for feature extraction and anomaly detection [29,30]. By incorporating these two techniques, our proposed method can effectively detect abrupt changes in the dataset and provide a valuable tool for neural networks to trigger relearning modeling processes.

The methodology can be succinctly described as follows: Initially, the sliding window moves as the data stream slides to keep the latest data processed. Within each window, the kernel density estimation technique is applied to assess the probability density distribution of the data. This allows for the acquisition of a probability density distribution specific to each window. Subsequently, the information entropy for each window is computed alongside the first-order differential of the entropy values. In this process, we can observe that the entropy difference of the window containing the jump points will show a significant change. By analyzing the entropy difference, we can determine the sliding window that contains the jump points. Ultimately, with the detection of a window with jump points, statistical measures like mean and variance are employed to pinpoint the exact location of these jump points by analyzing the change in the mean and variance of the data.

Figure 4 presents the schematic diagram of the sliding window algorithm. The sliding window frames the time series by specifying a fixed-length window and advances in the specified direction with a fixed length and step size.

For an input variable, a set of sample data is obtained as follows:

$$u_1 = \{u_1, u_2, \cdots, u_l\}. \tag{7}$$

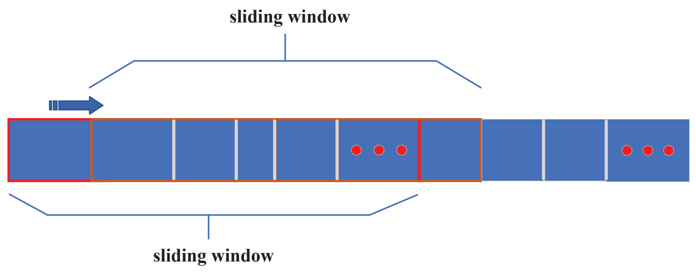


**Figure 4.** Schematic diagram of sliding window algorithm.

The sliding window's starting point is positioned at the initial point of the time series, yielding a time series $u_i^1$ with a length of $\omega$. This can be represented as follows:

$$u_i^1 = (u_1, u_2, \cdots, u_\omega). \tag{8}$$

Then, sliding window is subsequently moved one position forward along the time series, yielding a subsequent time series $u_i^2$, illustrated as follows:

$$u_i^2 = (u_2, u_3, u_4, \cdots, u_{\omega+1}). \tag{9}$$

In general terms, the sliding window will position the starting point of the sliding window at the $q$ th point of the time series to obtain a time series $u_i^q$ whose length is $\omega$, which can be expressed as follows:

$$u_i^q = \{u_q, u_{q+1}, \cdots, u_{q+\omega-1}\}. \tag{10}$$

Then, the distribution of data in each window can be calculated using the kernel density estimation method [31,32]. The kernel density estimation method is a common non-parametric estimation method. The process of kernel density estimation is to assign the probability of each point to its subordinate interval, and the final density function should be the sum of the density of all independent points. The kernel density estimation of the overall density function at any point can be expressed as follows:

$$P(u) = \frac{1}{\omega h}\sum_{i=1}^{\omega} K(\frac{u_i - u}{h}), \tag{11}$$

where $K(\cdot)$ is kernel function, $h$ represents window width, $u_i, i = 1, 2, \cdots, \omega$ and represents $\omega$ samples within each window. After obtaining the probability density function from Equation (11), information entropy then can be calculated as follows:

$$E_i = -\sum P(u_i)\log_2 P(u_i), \tag{12}$$

where $E_i, i = 1, 2, \cdots, g$, represents entropy for each sliding window. By calculating the information entropy for each sliding window, an entropy sequence $E$ can be obtained as follows:

$$E = \{E_1, E_2, \cdots, E_g\}. \tag{13}$$

The first-order differential calculation of the entropy sequence yields the difference sequence $E_d$, which is given by

$$E_{d_i} = E_{i+1} - E_i, \tag{14}$$

$$E_d = \left\{E_{d_1}, E_{d_2}, \cdots E_{d_{g-1}}\right\}, \tag{15}$$

where $E_{d_i}$ represents the change in entropy between adjacent windows. From the definition of information entropy. It can be seen that the first-order difference value of the information entropy of the sliding window containing the jump points is significantly higher than the first-order difference value of the information entropy of the sliding window without the jump points. By comparing the results of the first-order difference in entropy, the sliding window containing the jump points can be detected. Denote the window which contains the jump point as follows:

$$u_E = \{u_{q+1}, u_{q+2}, \cdots, u_{q+\omega}\}, \tag{16}$$

and for the data in the above window, we select the $q+d$ th point of the time series as the initial split point, where, $d = 1, 2, \cdots \omega$, the corresponding value is $u_{q+d}$, and the data in the window is divided into two parts,

$$u_{E1} = \left\{u_{q+1}, u_{q+2}, \cdots, u_{q+d}\right\}, \tag{17}$$

$$u_{E2} = \left\{u_{q+d+1}, u_{q+d+2}, \cdots, u_{\omega}\right\}, \tag{18}$$

and the degree to which any data $u_{q+r}, r = 1, 2, \cdots, \omega$ within the window deviates from the empirical estimate is expressed as follows:

$$e_r = \frac{|u_{q+r} - \mu_i|}{\sigma_t}, \tag{19}$$

where $\mu_i$ and $\sigma_i$ are the mean and standard of the two parts, respectively. To compute the cumulative deviation of all points encompassed by the sliding window, the calculation can be expressed as follows:

$$e_{sum} = \sum_{r=1}^{\omega} e_r. \tag{20}$$

The optimal split point is determined by shifting the initial position until the $e_{sum}$ reaches its minimum value. Finally, the starting and ending positions of the jump dataset are obtained, so that the boundary of the input data is detected and stored, and different input and output data sets are divided.

In summary, the steps of the data detection algorithm based on sliding window and information entropy can be summarized as follows:

Step 1: Use slide window to intercept the time series;

Step 2: Apply kernel density estimation method to esitimate the probability distribution for each window;

Step 3: Calculate the information entropy of the data in each window to find the first-order difference;

Step 4: Evaluate the degree of deviation of the data in each window from the empirical estimates, and obtain the window containing the jump dataset;

Step 5: Determine the start and end positions of the jump dataset;

Step 6: Obtain different boundary ranges in the statistics and divide the corresponding datasets.

After the boundary range of the input data is detected, the next part will be the construction of the event-triggered conditions for model relearning.

## *3.3. Neural Networks for Triggered Relearning*

The concept of the event-triggered mechanism dates back to 1959, and the event-triggered mechanism can be regarded as a non-isoperiodic triggered method of "execution on demand". The core idea is to determine whether to trigger the current operation by defining a reasonable "event" generator and judging whether the "event" has occurred. If the boundary range of the prediction dataset and the modeling training input data is inconsistent, the established neural network will generate large errors, and then errors are redefined to construct an event-triggered condition group as the basis for triggering retraining. The flow of an event-triggered relearning modeling strategy is shown in Figure 5.

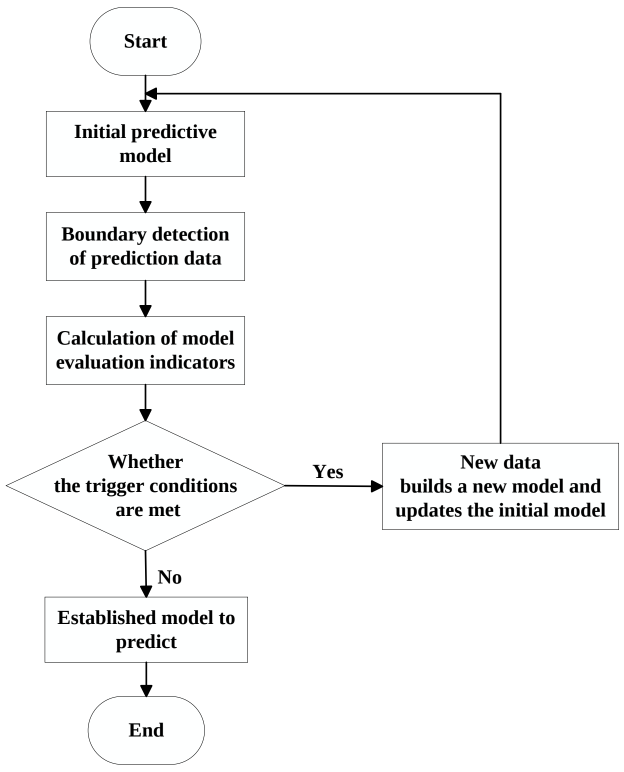

Start
Initial predictive model
Boundary detection of prediction data
Calculation of model evaluation indicators
Whether the trigger conditions are met
Yes
New data builds a new model and updates the initial model
No
Established model to predict
End

**Figure 5.** Flow chart of event-triggered relearning modeling.

The neural network established by Equation (5) satisfies the constraints of the boundary range of its modeling training input data under stationary conditions. Therefore, accurate predictions can be made on the data, and the predicted values of the neural network are given by

$$\widehat{y}(k) = f^1_{NN}(u(k)) s.t. u(k) \in D_1, \tag{21}$$

where $u(k)$ is the input data of the neural network, $D_1$ is the boundary range of the input data, and $\widehat{y}(k)$, $y(k)$ are the prediction value and the real value, respectively. Mean square error (MSE) and mean absolute error (MAE) can be used to represent the model performance. Mean squared error (MSE) is defined as follows:

$$MSE = \frac{1}{N}\sum_{k=1}^{N}[y(k) - \widehat{y}(k)]^2, \tag{22}$$

while mean absolute error (MAE) is defined as follows:

$$MAE = \frac{1}{N}\sum_{k=1}^{N}|y(k) - \widehat{y}(k)|, \tag{23}$$

where $N$ is the number of output variables. The MSE and MAE serve as model evaluation metrics, assisting in decision-making and model refinement to better align with real-world application requirements. When the boundary range of the predicted data is $D_2$ and exceeds the boundary range $D_1$ of the modeling training input data , the neural network established within the boundary of $D_1$ predicts the data of the boundary range of $D_2$, which can be expressed as follows:

$$\widehat{y}(k) = f^1_{NN}(u(k)) s.t. u(k) \in D_2. \tag{24}$$

The error between the true value and the prediction value, is given by

$$e = y(k) - \widehat{y}(k), \tag{25}$$

and the established neural network model cannot accurately predict data that falls outside the boundaries of the modeled training input.

When the system operates under a nonstationary condition, there is a jump dataset in the prediction data, often involving a wider range of data points and having a significant impact on the entire dataset or a specific group. If the original model continues to predict the jump point, from the first jump point on, the subsequent predictions will produce large errors. These errors can be much greater than the prediction errors of the data under stationary conditions. This situation indicates a significant reduction in the predictive performance of the model. Therefore, when the prediction error reaches a certain threshold, the model needs to be relearned. Consequently, we define and employ both the current error and the cumulative error as criteria to initiate the relearning process. To be specific, the current error represents a measure of deviation or difference between the prediction value at the jump point and the true value at a given point in time, while cumulative error is a measure of the overall deviation of the prediction values and the true values from the occurrence of the jump point and multiple subsequent jump points over a period of time. When the jump data is at the time point $k = k_k$, the current error is given by the following description,

$$e_{cur} = y(k_k) - f_{NN}^{1}(u(k_k)), \tag{26}$$

where $y(k_k)$, $f_{NN}^{1}(u(k_k))$ are true value and prediction value at the time point $k = k_k$, respectively. And, the cumulative error across $L - k_k + 1$ to $L$ groups of data following the jump point is defined as:

$$e_{cum} = \frac{1}{L - k_k + 1} \sum_{k=k_k}^{L} (y(k) - f_{NN}^{1}(u(k)))^2. \tag{27}$$

Setting up the threshold $e_{T1}$ and threshold $e_{T2}$, the event-triggered condition group then can be established as follows:

$$\begin{cases} e_{cur} \geq e_{T1} \\ e_{cum} \geq e_{T2} \end{cases}, \tag{28}$$

where threshold $e_{T1}$ stands for the average of prediction errors for data within the boundary range of its training input data, and threshold $e_{T2}$ represents the MSE of prediction values for data within the boundary range of its training input data.

Defining and calculating current and cumulative errors with set thresholds helps pinpoint abnormal data points and their impact on the dataset. Once event-triggered conditions are satisfied, the corresponding nonlinear mapping relationship $f_{NN}^{2}$ is established by retraining the dataset under the new boundary $D_2$, which is given as follows:

$$Y(k) = f_{NN}^{2}(U(k)) s.t. U(k) \in D_2. \tag{29}$$

When the boundary range of the prediction data aligns with that of the training input data for the established model, the event-triggered conditions are not activated. In such cases, the system can directly transition to the pre-established model corresponding to the prediction data's boundary range, bypassing the need for additional training and learning. Finally, we can get a one-to-one correspondence between the boundary range of the input data and the neural network model, and the corresponding representation can be expressed as follows:

$$Y(k) = \begin{cases} f_{NN}^{1} U(k), & U(k) \in D_1 \\ f_{NN}^{2} U(k), & U(k) \in D_2 \\ \vdots & \\ f_{NN}^{j} U(k), & U(k) \in D_j \end{cases}, \tag{30}$$

where $D_1, D_2, \cdots, D_j$ represent the different boundary ranges of the forecast data, respectively. $f^1_{NN}, f^2_{NN}, \cdots, f^j_{NN}$, are different prediction models corresponding to the predicted data of different boundaries, respectively. In comparison with the traditional BP neural network, our proposed method can accurately predict data beyond the boundary of the modeling training input data. Furthermore, compared with the traditional online random weight network, the proposed method can more accurately predict data characterized by large distribution differences while also reducing computational cost.

The proposed modeling strategy for BP neural network relearning based on event-triggered mechanism is given by Algorithm 1.

**Algorithm 1:** The triggered relearning algorithm.

```
1  Initialize the network $f^1_{NN}$ and boundary of input data $D_1$ ;
2  According to Equations (22) and (23), calculate MSE and MAE, set $e_{T1}$ and $e_{T2}$ ;
3  for $D_i$, $i = 1$ to $j$ do
4      if $D_{i+1} \notin D_i$ and satisfy conditions by Equation (28) then
5          retrain data corresponding to boundary $D_{i+1}$ and obtain new network $f^{i+1}_{NN}$
             by Equation (29);
6      else
7          switch directly to established network $f^i_{NN}$ corresponding to $D_i$ by
             Equation (30);
8      end
9  end
10 return result
```

## 4. Numerical Simulation

In order to verify the effectiveness of the proposed algorithm, the following numerical simulation experiments are carried out. A system description, the Hammerstein model, a synergistic combination of a nonlinear component followed by a linear one, is employed to test the effectiveness of the proposed algorithm [33–35]. This model provides a more precise simulation of specific systems and is widely adopted in real-world applications, as demonstrated in Figure 6.

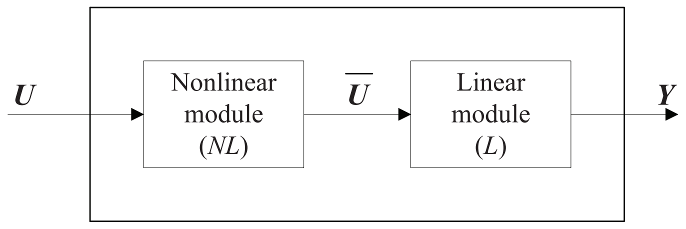


**Figure 6.** Hammerstein model.

$U$, and $\overline{U}$, $Y$ in Figure 6 are system inputs, intermediate variables, and outputs, respectively. $NL$ and $L$ represent nonlinear modules and linear modules, respectively. The Multi-Input Multi-Output (MIMO) Hammerstein system is more complex and able to describe a variety of nonlinear systems with relative accuracy, such as system models for hydraulic excavators, solid oxide fuel cells, and magnetorheological dampers. By introducing a static nonlinear function at the input stage of the state–space model, we can construct a MIMO Hammerstein model with Gaussian noise. This stochastic system can be formulated as follows:

$$\overline{U}(k) = f[U(k)], \tag{31}$$

$$X(k+1) = AX(k) + B\overline{U}(k) + W(k), \tag{32}$$

$$Y(k) = CX(k), \tag{33}$$

where $U(k) \in R^m$, $X(k) \in R^p$ , $Y(k) \in R^n$ are the input, the state vector, the output of the system at the discrete time $k$, sequentially. Matrices $A, B, C$ are system matrix, input matrix, output matrix, respectively. $f(\cdot)$ is the nonlinear change, $\overline{U}(k) \in R^m$ is the intermediate variable after nonlinear change, and $W(k)$ is Gaussian noise. The input is processed by a nonlinear module which is expressed as follows:

$$\overline{U}(k) = f[U(k)] = [\overline{u}_1(k), \overline{u}_2(k), \cdots, \overline{u}_m(k)]^T. \tag{34}$$

In this paper, the nonlinear part can be expressed by a polynomial as follows:

$$\begin{array}{c} \overline{u}_1(k) = \gamma_{11}u_1(k) + \gamma_{12}u_1^2(k) + \cdots + \gamma_{1m}u_1^m(k) \\ \overline{u}_2(k) = \gamma_{21}u_2(k) + \gamma_{22}u_2^2(k) + \cdots + \gamma_{2m}u_2^m(k) \\ \vdots \\ \overline{u}_m(k) = \gamma_{m1}u_m(k) + \gamma_{m2}u_m^2(k) + \cdots + \gamma_{mm}u_m^m(k) \end{array}, \tag{35}$$

where polynomial coefficients $\Theta$ for the nonlinear part can be defined as follows:

$$\Theta = \begin{bmatrix} \gamma_{11} & \cdots & \gamma_{1m} \\ \vdots & \ddots & \vdots \\ \gamma_{m1} & \cdots & \gamma_{mm} \end{bmatrix}. \tag{36}$$

Thus the Hammerstein system, $f_j(\cdot), j = 1, 2, \cdots$, represents a different nonlinear transformation. The coefficients by different polynomials $\Theta_j$ are given as follows:

$$\Theta_j = \begin{bmatrix} \gamma_{11}^j & \cdots & \gamma_{1m}^j \\ \vdots & \ddots & \vdots \\ \gamma_{m1}^j & \cdots & \gamma_{mm}^j \end{bmatrix}. \tag{37}$$

When the parameter $\Theta_j$ in Equation (37) varies, it alters the nonlinear module, which in turn changes the entire system. Simultaneously, the range of the system's inputs also shifts due to this change, leading to a corresponding variation in the output, which can be demonstrated in Equation (38),

$$f_j[U(k)] = \begin{cases} \overline{u}_1^j(k) = \gamma_{11}^j u_1 + \ldots + \gamma_{1m}^j u_1^m, \\ \qquad u_1 \in [a_1, b_1] \\ \overline{u}_2^j(k) = \gamma_{21}^j u_2 + \ldots + \gamma_{2m}^j u_2^m, \\ \qquad u_2 \in [a_2, b_2] \\ \qquad\qquad \vdots \\ \overline{u}_m^j(k) = \gamma_{m1}^j u_m + \ldots + \gamma_{mm}^j u_m^m, \\ \qquad u_m \in [a_m, b_m] \end{cases}, \tag{38}$$

where $a_i, b_i, i = 1, 2, \cdots, m$, represent the lower and upper bounds of the range of values for each input variable, respectively.

### *4.1. Acquisition of Experimental Data*

In this paper, matrices $A, B$ and $C$ in the Hammerstein model in Equation (32) are given as follows:

$$A = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.2 \end{bmatrix}, C = \begin{bmatrix} 0.3 & 0 \\ 0.1 & 0.6 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.1 & -0.1 & 0.2 & 0 & -0.3 \\ 0.3 & -0.2 & 0 & 0.1 & 0 \end{bmatrix}.$$

Altering the coefficients of the polynomial in the nonlinear part results in corresponding changes in the system. Given three different sets of polynomial coefficients, three different nonlinear Hammerstein systems are obtained, with the following polynomial coefficients:

$$\Theta_1 = \begin{bmatrix} 0.309 & 0.192 & 0 & 0 & 0 \\ 0.289 & 0.442 & 0 & 0 & 0 \\ 0.533 & 0.617 & 0.272 & 0 & 0 \\ 0.705 & 0.421 & 0.127 & 0.098 & 0 \\ 1.732 & 0.896 & 0.141 & 0.057 & 0.604 \end{bmatrix},$$

$$\Theta_2 = \begin{bmatrix} 0.299 & 0.186 & 0 & 0 & 0 \\ 0.291 & 0.438 & 0 & 0 & 0 \\ 0.531 & 0.596 & 0.265 & 0 & 0 \\ 0.671 & 0.394 & 0.118 & 0.087 & 0 \\ 1.658 & 0.771 & 0.136 & 0.069 & 0.058 \end{bmatrix},$$

$$\Theta_3 = \begin{bmatrix} 0.409 & 0.208 & 0 & 0 & 0 \\ 0.311 & 0.507 & 0 & 0 & 0 \\ 0.533 & 0.617 & 0.272 & 0 & 0 \\ 0.801 & 0.439 & 0.206 & 0.011 & 0 \\ 1.859 & 0.901 & 0.157 & 0.068 & 0.077 \end{bmatrix}.$$

$W(k)$ in Equation (32) is Gaussian white noise sequence with variance of $\sigma_v^2 = 0.01$ and a null mean.

A dataset I containing 1000 samples with 5 inputs and 2 outputs can be obtained by processing each input component $u_i(k)$ in the nonlinear Hammerstein system with Gaussian white noise corresponding to the parameter $\Theta_1$ , with null mean and a variance of 1. Similarly, datasets II and III can be obtained by multiplying each input component of the Hammerstein system with a constant $k1$ and $k2$ , respectively, with null mean and a variance of 1. Dataset II and III also contain 1000 samples with five inputs and two outputs. Repeating the above operation, dataset IV, dataset V, and dataset VI with five inputs and two outputs, and the length of 1000 can be obtained, respectively. Among them, the input data of dataset I and IV, II and V, III and VI have the same boundary range, respectively.

### *4.2. Triggered Relearning for Neural Network*

#### 4.2.1. Model Development

The above datasets I, II, III are combined into a dataset with a length of 3000, where the input data contain three different boundary ranges. The boundary range of the input component $u_1$ is detected and divided, as shown in Figure 7. Figure 7's first graph shows entropy trends, highlighting significantly higher values for the fifth and tenth windows containing jump points compared to others. The second graph depicts the sharp changes in entropy's first-order difference for windows with jump points. The third graph uses Equations (19) and (20) to identify windows with jump points, and the fourth illustrates using a data detection algorithm to locate the start of the jump dataset using sliding windows and information entropy.

Datasets I, II, and III have unique boundary ranges $D_1$, $D_2$, and $D_3$. Model I predicts within $D_1$ effectively, but accuracy falters when dataset II's boundary $D_2$ extends beyond $D_1$. The neural network initiates relearning when boundaries shift, using prediction errors to set event-triggered conditions. For boundary $D_2$, Model II retrains using dataset II. If dataset III's boundary $D_3$ exceeds $D_2$, a similar retraining for Model III occurs using dataset III. A total of 700 groups of data in each dataset are trained, and 300 groups of data are predicted, where the prediction effect of the triggered relearning method on the three pieces of data is shown in Figure 8. In Figure 8, the blue and red lines represent the prediction values of BP-triggered relearning and true values, respectively. We can see that the prediction value is very close to the real value.

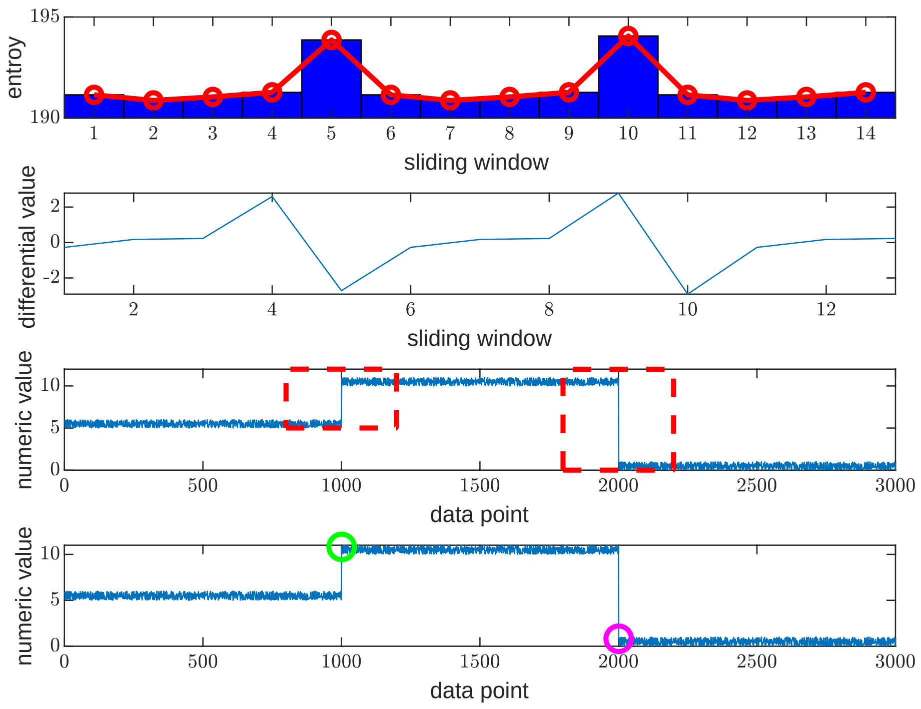


**Figure 7.** Jump dataset detection of input data. The red circles in the first graph represent the entropy of each window. The solid line in the second graph indicates the change in the first-order difference value of entropy. The dotted line in the third graph implies sliding windows with jump points. The green and purple circles in the fourth figure show the starting jump points in the data, respectively.

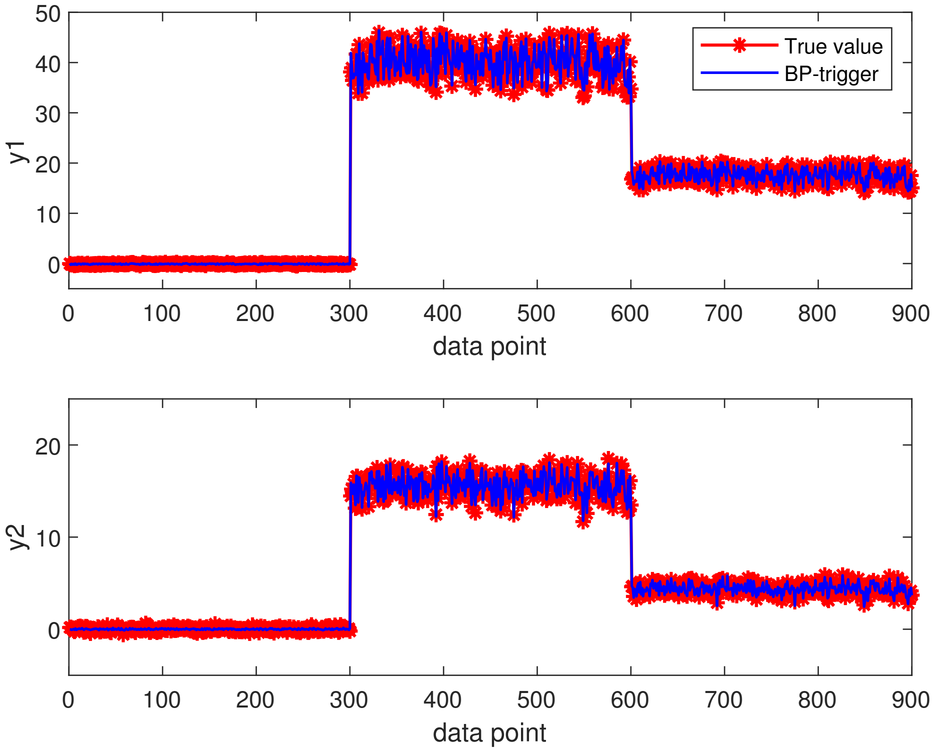


**Figure 8.** Prediction effect diagram of triggering relearning.

If the input data for Dataset IV matches the $D_1$ boundary range of Dataset I, the system defaults to Model I for direct prediction. Similarly, identical boundary ranges for new input data with Dataset II or III trigger the use of Model II and III, respectively, for direct predictions of Dataset V and VI. By using 300 sets of data in datasets I–VI to form a dataset containing six input data with different boundary ranges, the prediction effect of the triggered relearning method is shown in Figure 9. In Figure 9, the blue and red lines represent the prediction values of BP-triggered relearning and true values, respectively. It can be seen that the prediction value of the triggered relearning model is very close to the true value, and the distribution of the error probability density curve is consistent with the Gaussian distribution curve around the null mean.

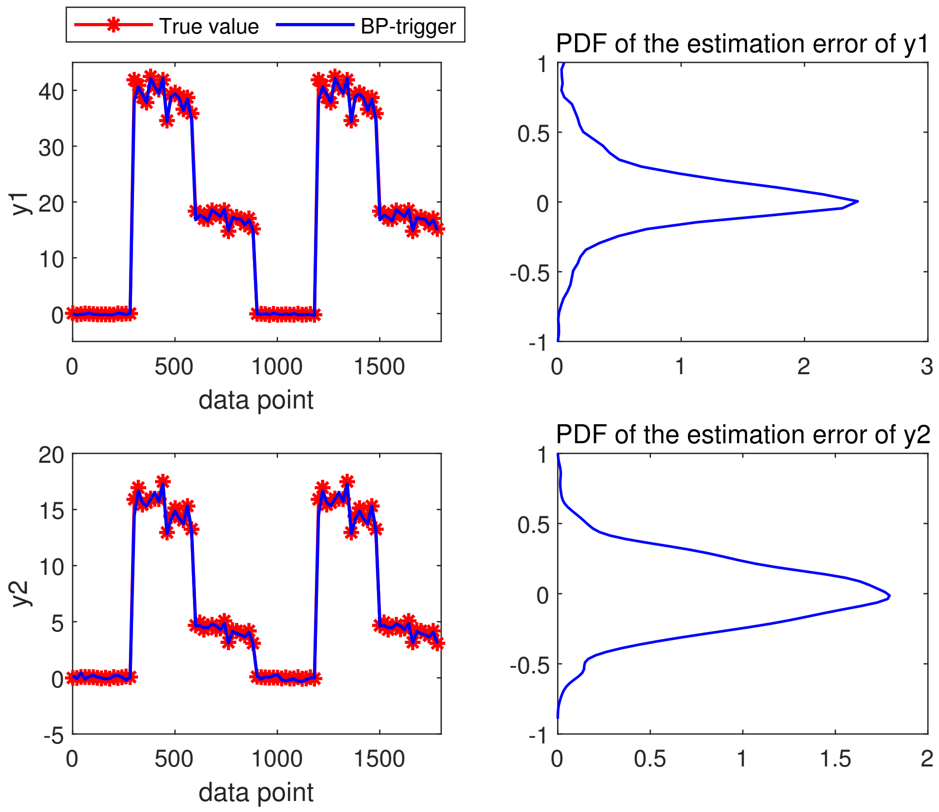


**Figure 9.** Estimation effect and error of triggered strategy.

### 4.2.2. The Effect of Model Estimation

Following the procedure introduced above, 300 sets of data are taken from data sets I–V I to form an 1800 groups test dataset. We use the BP neural network for the triggered relearning modeling method to predict the input and output datasets containing the jump dataset. In order to show the superiority of the proposed method, the conventional BP neural network and the online sequential random weight neural network (OS-RVFLNs) modeling method also applied to the same system to compare. To ensure the validity of the comparative experiment, the training data and test data are exactly the same, and are all datasets containing the jump dataset. The modeling prediction results are shown in Figure 10.

In Figure 10, the blue line, green line and black line represent the prediction values of BP neural networks for triggered relearning, conventional BP neural networks, and OS-RVFLNs, respectively. For conventional neural networks, if the data structure changes, timely adjustments will not be made, so the prediction errors will be very large. Traditional online sequential random weight neural network has a certain degree of difference in the prediction effect of different boundary data. Triggered relearning effectively leverages new data to address the limitations of conventional online learning algorithms when confronted with substantial disparities in the distribution of new and old data. By the timely updating of the model parameters, it enhances the accuracy and generalizability of the model, aligning it more closely with the characteristics and distribution of the new data. In addition, that which is no longer representative or crucial can be filtered out, allowing the focus to be on data with high reliability and significance. When the boundary range of the prediction data aligns with that of the modeling training input data, the system can seamlessly switch to the corresponding model for data prediction within that boundary, thereby reducing computational costs. The MSE and MAE of the prediction results from different methods are presented in Table 1. From Table 1, we can see that the MSE and MAE of the BP neural network for triggered relearning are the smallest. This result demonstrates that the predictive performance of this model outperforms several other methods, marking it as the most effective in the comparison group. Furthermore, the number of float point operations is much smaller than that of the online random weight method, indicating that the calculation amount of this method is smaller.

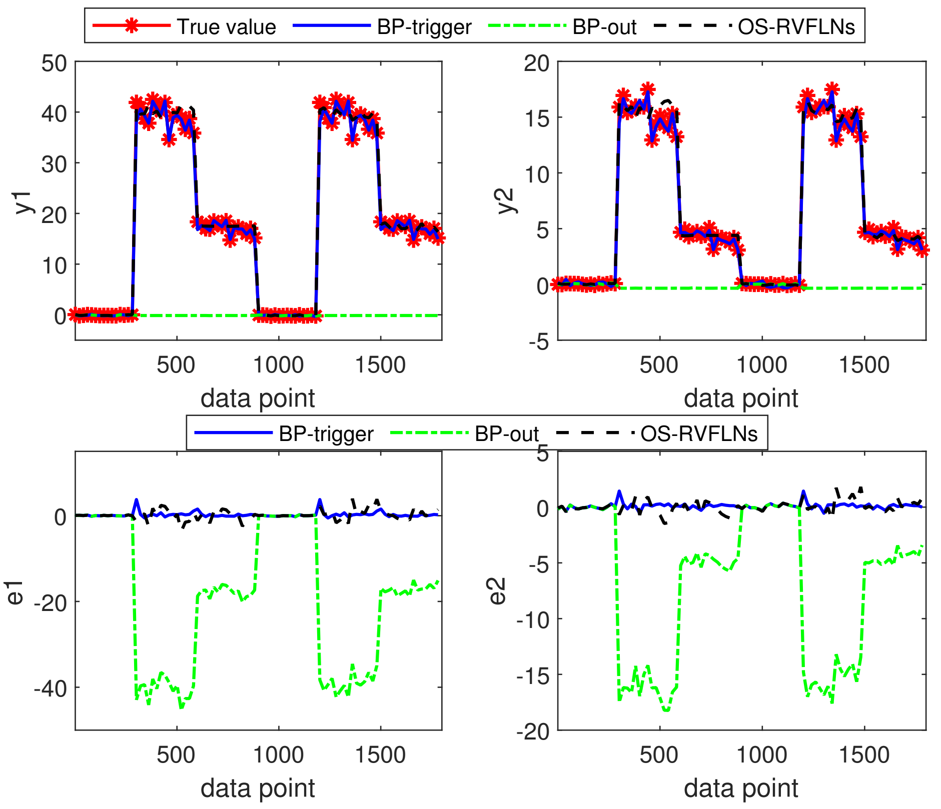


**Figure 10.** Estimation performance and error of different modeling methods.

**Table 1.** Evaluation of performance indicators for different models.

| Model | FLOPs | MSE | | MAE | |
|---|---|---|---|---|---|
| | | Y1 | Y2 | Y1 | Y2 |
| BP | 148,180 | 81.23 | 74.93 | 5.71 | 6.26 |
| OS-RVFLNs | 3,413,100 | 0.18 | 0.12 | 0.26 | 0.27 |
| BP-trigger | 1,326,180 | 0.09 | 0.05 | 0.19 | 0.18 |

To facilitate a clear and intuitive assessment of the strengths and weaknesses of these methods, the error and their probability density function curves of several modeling prediction models are generated, as shown in Figure 11, respectively. It can be seen that for both outputs, the conventional BP neural network and OS-RVFLNs are wider and shorter, indicating a worse prediction performance. Compared to the other two methods, the pdf of the prediction error generated by the proposed BP-triggered relearning method has a much sharper and taller shape, indicating that the proposed method engaged a better prediction job.

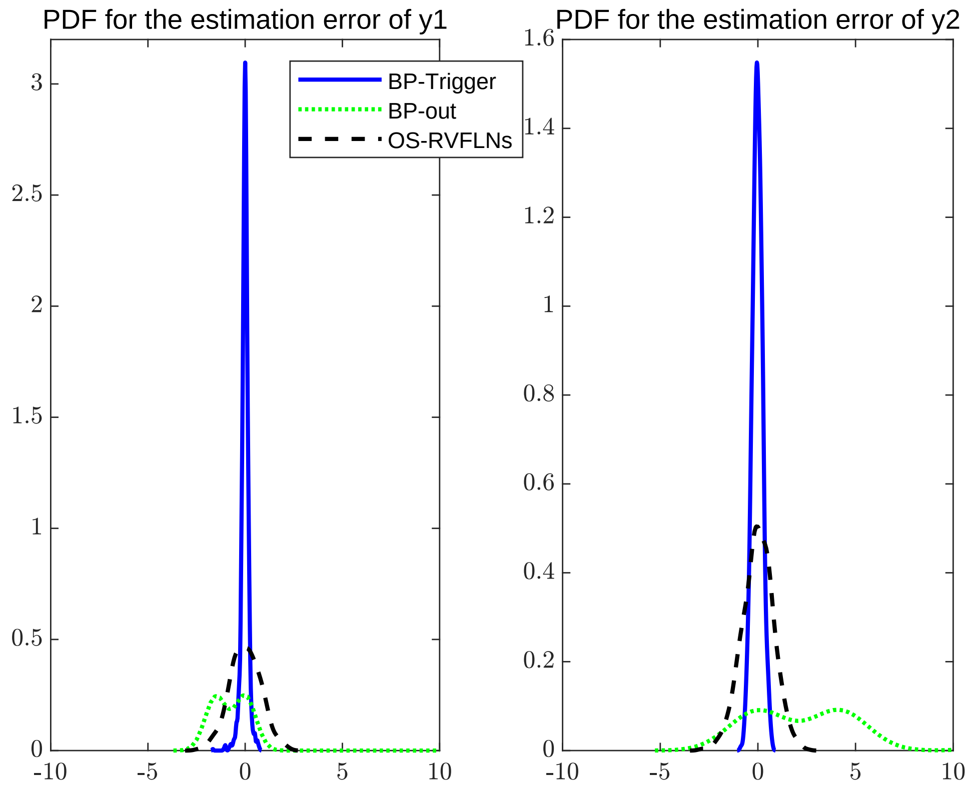


**Figure 11.** Error probability density distribution curves of different models.

## 5. Conclusions

To address the challenge of decreased model accuracy in conventional neural networks caused by the boundary range inconsistency between prediction data and modeling training input data, this paper has proposed a novel algorithm that involves detecting and segmenting the boundary range of prediction data using a combined approach of sliding window and information entropy. Through the combination of the event-triggered mechanism and the data-driven neural network modeling method, the prediction errors of the input data of different boundary ranges of the current model have been used to construct an event-triggered condition group and set the corresponding threshold. If the prediction error meets the triggered conditions, indicating a significant shift in system behavior, we will retrain the model using datasets that reflect the new operational boundaries. If the prediction data with the same boundary range appears again, it will directly switch to the corresponding known model to predict. In this way, we can realize the one-to-one correspondence between the boundary range of the prediction data and the neural network model. The simulation experiment applies the nonlinear Hammerstein model with Gaussian white noise to simulate the complex industrial stochastic processes. By comparing with traditional online learning algorithms and conventional neural network modeling methods, it has been concluded that the proposed method can better adapt to the changes of the boundary range of prediction data caused by non-stationary working conditions of complex systems. It has been proven that the proposed method can improve not only the accuracy of prediction, but also the adaptability of the model. Furthermore, the demands for the computing and storage resources of the algorithm have been alleviated, and the scalability and efficiency of the algorithm have also been enhanced.

**Author Contributions:** Conceptualisation, Y.Z. (Yong Zhang); methodology, Y.Z. (Yuyang Zhou); validation, J.L.; funding acquisition, J.C.; writing—original draft preparation, J.L.; resources, Y.Z. (Yong Zhang); writing—review and editing, Y.Z. (Yuyang Zhou). All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported in part by the National Natural Science Foundation of China under Grant 62263026, Grant 61973137, and Grant 62373165, and in part by the Fundamental Research Funds for Inner Mongolia University of Science and Technology under Grant 2024QNJS003, and in part by the Natural Science Foundation of Jiangsu Province Grant BK20201339.

**Data Availability Statement:** The data will be made available by the authors on request.

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

## References

1. Zhou, Y.; Herzallah, R. Probabilistic message passing control for complex stochastic switching systems. *J. Frankl. Inst.* **2021**, *358*, 5451–5469. [CrossRef]
2. Sykora, H.; Sadeghpour, M.; Ge, J. On the moment dynamics of stochastically delayed linear control systems. *Int. J. Robust Nonlinear Control* **2020**, *30*, 8074–8097. [CrossRef]
3. Herzallah, R. A fully probabilistic design for stochastic systems with input delay. *Int. J. Control* **2021**, *94*, 2934–2944. [CrossRef]
4. Gu, Z.; Lv, D.; Li, X.; Zhang, Y.; Dai, X. Fusion prediction of blast furnace temperature based on the combination of knowledge and data. *China Meas. Test* **2022**, *48*, 1–9.
5. Zhuang, T.; Yang, C. Silicon content forecasting method for hot metal based on Elman-Adaboost strong predictor. *Metall. Ind. Autom.* **2017**, *41*, 1–6+17.
6. Cui, G.; Chen, R.; Yu, K. The Study on Temperature Prediction of ELM Furnace Based on Multiscale Decomposition. *Control Eng.* **2020**, *27*, 1901–1906.
7. Jiang, Z.; Dong, M.; Gui, W.; Yang, C. Two-dimensional Prediction for Silicon Content of Hot Metal of Blast Furnace Based on Bootstrap. *Acta Autom. Sin.* **2016**, *42*, 715–723.
8. Zhou, P.; Zhang, L.; Li, W. Autoencoder and PCA Based RVFLNs Modeling for Multivariate Molten Iron Quality in Blast Furnace Ironmaking. *Acta Autom. Sin.* **2018**, *44*, 1799–1811.
9. Su, X.; Zhang, S.; Yin, Y.; Hui, Y.; Xiao, W. Prediction of hot metal silicon content for blast furnace based on multi-layer online sequential extreme learning machine. In Proceedings of the 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 8025–8030.
10. Zhou, P.; Yuan, M.; Wang, H.; Wang, Z.; Chai, T. Multivariable dynamic modeling for molten iron quality using online sequential random vector functional-link networks with self- feedback connections. *Inf. Sci.* **2015**, *325*, 237–255. [CrossRef]
11. Jiang, Z.; Xu, C.; Gui, W.; Jiang, K. Prediction Method of Hot Metal Silicon Content in Blast Furnace Based on Optimal Smelting Condition Migration. *Acta Autom. Sin.* **2022**, *48*, 194–206.
12. He, X.; Ji, J.; Liu, K.; Gao, Z.; Liu, Y. Soft sensing of silicon content via bagging local semi-supervised models. *Sensors* **2019**, *19*, 3814. [CrossRef] [PubMed]
13. Yu, H.; Chen, T.; Hao, F. A new event-triggered control scheme for stochastic systems. *IEEE Trans. Autom. Control* **2022**, *68*, 1463–1478. [CrossRef]
14. Zhao, F.; Gao, W.; Jiang, Z.; Liu, T. Event-triggered adaptive optimal control with output feedback: An adaptive dynamic programming approach. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 5208–5221. [CrossRef] [PubMed]
15. Shanmugam, S.; Vadivel, R.; Gunasekaran, N. Finite-Time Synchronization of Quantized Markovian-Jump Time-Varying Delayed Neural Networks via an Event-Triggered Control Scheme under Actuator Saturation. *Mathematics* **2023**, *11*, 2257. [CrossRef]
16. Li, G.; Zeng, J.; Liu, J. Effluent Quality-Aware Event-Triggered Model Predictive Control for Wastewater Treatment Plants. *Mathematics* **2023**, *11*, 3912. [CrossRef]
17. Liu, Y.; Zhu, Q. Event-triggered adaptive neural network control for stochastic nonlinear systems with state constraints and time-varying delays. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 1932–1944. [CrossRef]
18. Yu, H.; Chen, T. Event-triggered neural-network adaptive control for strict-feedback nonlinear systems: Selections on valid compact sets. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 4750–4762. [CrossRef]
19. Xing, L.; Wen, C. Dynamic event-triggered adaptive control for a class of uncertain nonlinear systems. *Automatica* **2017**, *62*, 2071–2076. [CrossRef]
20. Herzallah, R.; Zhou, Y. A fully probabilistic control framework for stochastic systems with input and state delay. *Sci. Rep.* **2022**, *12*, 7812. [CrossRef]
21. Wei, H. Stochastic Finite-Time Stability for Stochastic Nonlinear Systems with Stochastic Impulses. *Control Theory Appl.* **2022**, *14*, 817.
22. Ma, K.; Kumar, P. Incentive Compatibility in Stochastic Dynamic Systems. *IEEE Trans. Autom. Control* **2020**, *66*, 651–666. [CrossRef]
23. Wu, Y.; Feng, J. Development and application of artificial neural network. *Wirel. Pers. Commun.* **2018**, *102*, 1645–1656. [CrossRef]
24. Abraham, T. (Physio) logical circuits: The intellectual origins of the McCulloch–Pitts neural networks. *J. Hist. Behav. Sci.* **2002**, *38*, 3–25. [CrossRef]

25. Yin, L.; Li, L.; Jiang, Z. Prediction of silicon content in hot metal using neural network and rough set theory. *J. Iron Steel Res.* **2019**, *31*, 689–695.
26. Liang, W.; Wang, G.; Ning, X.; Zhang, J.; Li, Y.; Jiang, C.; Zhang, N. Application of BP neural network to the prediction of coal ash melting characteristic temperature. *Fuel* **2020**, *260*, 116324. [CrossRef]
27. Zhuo, Y.; Zhou, X.; Wang, Q.; Zou, Z.; Wang, S.; Liu, Y. Research on a Sliding Window Bats Code Method. In Proceedings of the 2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China, 18–20 December 2020; pp. 404–409
28. Xie, Y.; Yang, T.; Jia, D.; Zhang, C.; Li, Y. Minor fault detection based on PCA and moving window cumulative sum. *Comput. Appl. Softw.* **2023**, *40*, 60–66+96.
29. Sabirov, D. Information entropy of mixing molecules and its application to molecular ensembles and chemical reactions. *Comput. Theor. Chem.* **2020**, *1187*, 112933. [CrossRef]
30. Abdelmonem, M. Information entropies for the Morse potential using the J-matrix method. *Results Phys.* **2017**, *7*, 1178–1180. [CrossRef]
31. Wang, Z. *Research and Modeling Application of VSG Method Based on Monte Carlo and Kernel Density Estimation*; Beijing University of Chemical Technology: Beijing, China, 2021.
32. Pan, J.; Zou, Z.; Sun, S.; Su, Y.; Zhu, H. Research on output distribution modeling of photovoltaic modules based on kernel density estimation method and its application in anomaly identification. *Sol. Energy* **2022**, *235*, 1–11. [CrossRef]
33. Houda, S.; Samira, K. A recursive parametric estimation algorithm of multivariable nonlinear systems described by Hammerstein mathematical models. *Appl. Math. Model.* **2015**, *39*, 4951–4962.
34. Liu, Q. *Recursive Identification for Hammerstein-Wiener Nonlinear Systems with Non-Uniform Sampling*; Jiangnan University: Wuxi, China, 2022.
35. Zheng, T.; Li, F.; Luo, Y.; Liu, R.; Gu, Y. Review of neural network-based methods for solving partial differential equations. *Control Eng. China* **2022**, *29*, 2034–2041.