Multivariate Time-Series Prediction in Industrial Processes via a Deep Hybrid Network under Data Uncertainty

Yao, Yuantao; Yang, Minghan; Wang, Jianye; Xie, Min

Published in:
IEEE Transactions on Industrial Informatics

Published: 01/02/2023

Document Version:
Post-print, also known as Accepted Author Manuscript, Peer-reviewed or Author Final version

Publication record in CityU Scholars:
Go to record

Published version (DOI):
10.1109/TII.2022.3198670

Publication details:

Citing this paper
Please note that where the full-text provided on CityU Scholars is the Post-print version (also known as Accepted Author Manuscript, Peer-reviewed or Author Final version), it may differ from the Final Published version. When citing, ensure that you check and use the publisher's definitive version for pagination and other details.

General rights
Copyright for the publications made accessible via the CityU Scholars portal is retained by the author(s) and/or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights. Users may not further distribute the material or use it for any profit-making activity or commercial gain.

Publisher permission
Permission for previously published items are in accordance with publisher's copyright policies sourced from the SHERPA RoMEO database. Links to full text versions (either Published or Post-print) are only available if corresponding publishers allow open access.

Take down policy
Contact lbscholars@cityu.edu.hk if you believe that this document breaches copyright and provide us with details. We will remove access to the work immediately and investigate your claim.

Download date: 14/09/2023
Multivariate Time Series Prediction in Industrial Processes via a Deep Hybrid Network under Data Uncertainty

Yuantaoyao, Member, IEEE, Minghan Yang, Jianye Wang, Min Xie, Fellow, IEEE

Abstract—With the rapid progress of the industrial Internet of Things (IIoT), reducing data uncertainty has become a critical issue in predicting the development trends of systems and formulating future maintenance strategies. This paper proposes an end-to-end, deep hybrid network-based, short-term, multivariate time series prediction framework for industrial processes. First, the maximal information coefficient (MIC) is adopted to extract the nonlinear variate correlation features. Second, a convolutional neural network (CNN) with a residual shrinkage module is designed to eliminate data uncertainty. Third, a bidirectional gated recurrent unit (Bi-GRU) network is connected in a time-distributed form to achieve step-ahead prediction. Last, an optimized Bayesian optimization method is adopted to optimize the model’s learning rate. A comparison with other state-of-the-art, deep learning (DL)-based, time series prediction methods in the case study illustrates the superiority of the proposed framework in noisy IIoT environments.

Index Terms—IIoT, multivariate time-series prediction, deep hybrid networks, hyperparameter optimization, data uncertainty

I. INTRODUCTION

REGARDLESS of the extent to which industrial systems have advanced with state-of-the-art techniques, economy, safety, and reliability have long remained fundamental and central topics of research [1], [2]. To this end, system managers and researchers usually select key metrics, such as the energy system’s power, electrical system’s load, and remaining useful life (RUL) in the mechanical system, to monitor system health and to develop maintenance policies [3], [4], [5]. Furthermore, these managers and researchers attempt to perform certain prediction tasks to develop improvement strategies. However, regardless of the metrics (short-term or long-term), the prediction tasks are inseparable from time series analyses.

With the development of information technology, the use of soft measurement technologies to monitor and analyze industrial systems is gradually increasing. Due to the complexity of physical-chemical reactions, heat-mass transfer, and thermal-hydraulic processes in most industrial process systems, such as petroleum, chemical, thermal, and nuclear power systems, it is difficult to establish a more precise mechanism model for operation state prediction [6]. Therefore, compared with the method based on the mechanism model, the data-driven time series prediction method is more suitable and can be better developed and applied to industrial process systems [7], [8].

Notable methods developed in the last decade include the support vector machine (SVM), the extreme learning machine (ELM), the genetic algorithm (GA), the artificial neural network (ANN), and data statistics methods related to the autoregressive integrated moving average (ARIMA), filter-based model, etc., which are widely defined as traditional machine learning (ML) algorithms [9], [10].

However, with the continuous development of big data and Industrial Internet of Things (IIoT) technology, inherent defects and limitations are exposed in traditional ML technology applied to processing high-dimensional big data and efficiently representing complex functions. In addition, designing an effective ML system requires significant data domain expertise.

Recently, deep learning (DL) describes models that use multiple layers to represent potential features at higher and more abstract levels. DL attempts to extract high-level abstract features through linear and nonlinear transformations, making it possible to address this problem. Four typical depth models are defined in the historical literature and are widely utilized in time series-related prediction tasks in various industrial process systems: encoder-decoder, convolutional neural network (CNN), deep belief network (DBN), and recurrent neural network (RNN). These models have been successfully applied in various prediction tasks.
The RNN is one of the popular forms for achieving high-performance prediction tasks, such as RUL prediction for aeroengines, as proposed by Miao [11]; solar power level prediction, as reported by Zheng [12]; industrial soft sensor prediction, as developed by Yuan [13]; and a method of long-term steam generator (SG) temperature prediction in nuclear power plants, as reported by Zio [14].

In addition, some researchers attempt to design a hybrid model with their own form or variant form to complete the prediction task. For example, Cheng et al. proposed a deep Bayesian network combined with one RNN, variant-gate, recurrent unit (GRU) to accomplish meteorological, time-series prediction in the IoT environment [15]. Wang et al. proposed a CNN-long short-term memory (LSTM) network for level prediction in electric power systems [16]. Zheng et al. used CNN and LSTM to learn 2D link information and maintain long-term memory for traffic flow prediction [17]. Essien et al. combined autoencoders, CNNs, and LSTM for the prediction task in metal packaging plants [18].

The above research can serve as a helpful reference for deep hybrid network-based prediction models. However, several points in the multivariate time-series prediction task should be further considered:

1) The large-scale monitoring data in IIoT increases irrelevant information between target indicators and different variates.
2) The specific noise generated by different component monitors during IIoT system operation introduces data uncertainty and reduces the prediction accuracy.
3) Redundant information collected in the IIoT is not appropriately utilized to benefit the confidence of the prediction task.

Some researchers have developed DL-based prediction methods for noise elimination. Zhang et al. used principal component analysis (PCA) to directly extract the multivariate features and to connect them to the hybrid CNN-GRU model [19]. Wang et al. proposed variational mode decomposition (VMD) combined with the GRU model [20]. Pei et al. supported the wind speed forecasting method based on LSTM using the empirical wavelet transform (EWT) [21].

The above studies provide a good reference for DL-based prediction methods. However, for noise elimination, certain studies directly disregard the impact of noise on the prediction of deep model training. Other studies use traditional methods to build a separate noise processing module coupled with a DL module. The complexity of the model is related to the number of subsets after signal decomposition, reducing the advantages of end-to-end model training in deep networks.

To address this problem, we propose a deep hybrid network-based, multivariate time series prediction framework for crucial, monitored, industrial process safety metrics, which fuses the data noise uncertainty elimination and time series prediction tasks. Novel contributions of this study include the following:

1) Maximal information coefficient (MIC)-based variate selection is adopted to filter out the irrelevant information between multivariate time series and indicators that must be predicted.
2) A CNN with a soft residual shrinkage module is designed and optimized to eliminate the data noise uncertainty under an end-to-end framework.
3) One optimized, Bayesian theory-based, hyperparameter optimization method is designed to improve the learning procedure of the proposed deep hybrid network structure.

The monitored time series data are input into this integrated, end-to-end hybrid model. Therefore, it can adaptively eliminate the uncertainty from the noise of different variables and the interference of irrelevant data. Further, we efficiently use the redundant information in the monitored data of the same type between adjacent nodes, improving the prediction performance.

The remainder of this paper is organized as follows: Section II describes the technical background. The proposed prediction framework is shown in Section III. Section IV introduces the details of the case study. The results of other models and optimization procedures are given in Section V. Section VI presents the conclusion and an outlook on future work.

II. TECHNICAL BACKGROUND

A. Problem formulation

The problem in this study is formulated mainly as multistep, time-series prediction. The input is a sequence group with temporal and multimodal characteristics. The output is a specific monitoring metric with time sequence characteristics and with practical significance, such as temperature [20], power [12], or load [5]. This sequence is one-dimensional. Fig. 1 and Formula (1) further illustrate this many-to-many prediction task.

Fig. 1. Illustration of the target prediction task.

\[ Y = (Y_{1,1}, Y_{2,1}, \ldots, Y_{n,2}) = f_2\left( \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d-1} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d-1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n-1,1} & x_{n-1,2} & \cdots & x_{n-1,d-1} \\ x_{n,1} & x_{n,2} & \cdots & x_{n,d-1} \end{bmatrix} \right) \]

where \( Y \) is the prediction output; \( f_2 \) represents the designed deep hybrid model; \( n \) is the current sample number; \( d \) and \( m \) represent the length and width, respectively, of the sliding window; \( I \) represents the time step of the input; and \( Z \) represents the forecast step.
To divide the time series group, the sliding window strategy (the green box in Fig. 1) is indispensable. The goal is to use a previously observed set of input sequences to predict a fixed-length sequence of future time series values. This approach is similar to establishing the mapping relationship between multiple multidimensional variable matrices and indicator sequences. Additionally, the inherent temporal correlation characteristics should be considered.

**B. Residual CNNs**

In our previous work, we applied an optimized CNN to perform fault diagnosis under various conditions in nuclear IIoT systems [22]. The training accuracy of the designed model with four convolutional layers was less than that with three convolutional layers. An increase in the number of layers increases the training error.

Rather than using several network layers to fit a hidden nonlinear mapping, it is easier for the network to learn its residuals. This premise is the underlying concept of residual CNNs (RCNNs) [23]. The basic structure of an RCNN is shown in Fig. 2.

We let \( H(x) \) denote the underlying mapping of the output of the entire network fitted by partially stacked layers, where \( x \) is the input to these layers. The residual is calculated by \( H(x) - x \). A model based on the residual learning method is employed to reverse this calculation, and the output vector \( y \) is calculated as follows:

\[
y = H(x) = F(x, [W_i]) + x \tag{2}
\]

where \( F(x, [W_i]) \) represents the learned residual mapping and \( x \) is the input vector. In the two-layer structure shown in Fig. 1, \( F = W_\sigma (W_x) \), where \( \sigma \) represents the rectified linear unit (ReLU) function.

In **Formula 2**, the dimensions of \( x \) and \( F \) must be equal. When the input or output channels change, a linear shortcut connection \( W_r \) will be executed by the shortcut connection to match the dimensions between them as follows:

\[
y = F(x, [W_i]) + W_r x \tag{3}
\]

If the residual is 0, applying identity mapping without adding a network layer will neither improve nor degrade the performance and avoid degradation problems. If the residual value is not equal to 0, then the network performance can still be improved by increasing the number of network layers.

**C. Gate recurrent units**

A GRU network is a type of RNN that has been widely adopted and optimized to process time-series data in recent studies on prediction problems. The GRU structure is a variant of the LSTM structure; it achieves the same performance as the LSTM structure at a much lower computational cost [24]. The inner structures of RNN, LSTM, and GRU are shown in Fig. 3.

Two important gate states (update gate \( A \) and reset gate \( B \)) are obtained from the input \( x_t \) of the current node and the last transmitted state \( h_{t-1} \). Certain information of the previous hidden state \( h_{t-1} \) is filtered out by the reset gate \( B \) and combined with the current input \( x_t \) to obtain the new hidden state \( h_t' \). The next hidden state \( h_t \) is then generated from the new hidden state \( h_t' \) and the previous hidden state \( h_{t-1} \), with relative proportions controlled by the update gate \( A \).

---

**III. PROPOSED PREDICTION FRAMEWORK**

This section introduces a designed deep hybrid network-based prediction framework. **Fig. 4** shows a schematic of the framework, which contains three modules: variate normalization and selection, residual shrinkage, and step-ahead prediction.
The proposed framework primarily uses information about prediction metrics in multivariate time series data and employs a residual structure to adaptively decrease noise uncertainty. The data characteristics in the IIoT systems under investigation are redundant information that is first filtered for correlation using the nonlinear correlation evaluation method, i.e., the MIC, then refined for noise using a residual module, and ultimately output using the bidirectional GRU (Bi-GRU) structure.

### A. Variate normalization and selection module

The original input of the multivariate time series collected from different sensors has a different evolutionary trend. It is necessary to normalize and eliminate irrelevant information. The prediction accuracy is improved, and worthless information is removed by filtering. The dimension of the input features is also reduced.

Of the nonlinear correlation evaluation methods, the MIC, which Reshef proposed in 2011, is effective when applied in feature selection for prediction and classification [25]. The principle of the MIC is that if there is a correlation between two parameter variables, this correlation can be extracted by dividing the two-dimensional distribution of both parameters by their individual distributions.

Assume that there are two time series as follows:

\[ A = [a_i, i = 1,2,\ldots,n], B = [b_i, i = 1,2,\ldots,n] \]

The mutual information (MI) between them is:

\[ MI = \sum_{a \in A, b \in B} p(a,b) \log \frac{p(a,b)}{p(a)p(b)} \]

where \( p(a, b) \) is the joint probability density of \( A \) and \( B \) and \( p(a) \) and \( p(b) \) are the marginal distribution densities of \( A \) and \( B \), respectively.

\( G(x,y) \) is then defined as a division of the two-dimensional distribution of \( A \) and \( B \) such that the ranges of \( A \) and \( B \) are divided into an \( x \) interval and a \( y \) interval, respectively. Based on the number of samples falling into each corresponding two-dimensional interval, the MI values between these two regions are calculated. The maximum of the MI values for each pair of \( x \) and \( y \), denoted by \( \max \{MI_{g(x,y)} \} \), is then obtained. The normalized result is as follows:

\[ m(x, y) = \frac{\max \{MI_{g(x,y)} \}}{\log \min \{x, y\}} \]

\[ MIC(x, y) = \max \{m(x, y)\} \]

where \( M(x,y) \) is the characteristic matrix. Typically, the size of the grid resolution of \( (x, y) \), which equals the product of \( x \) and \( y \), does not exceed \( 0.6 \).

The MIC-based variate selection is applied at the beginning of the entire hybrid network to filter the input information from different operating parameters, as shown in the yellow section in Fig. 4.

### B. Residual shrinkage module

The designed residual shrinkage module is based on the soft threshold (ST) method [26], which is commonly employed in signal processing to eliminate noise information. The main principle is to shrink a signal whose absolute value is greater than a certain threshold \( \alpha \), toward this threshold while eliminating signals whose absolute values are less than \( \alpha \). The formula for implementing an ST is as follows:

\[ y_{\alpha} = \begin{cases} x - \alpha & x > \alpha \\ 0 & -\alpha \leq x \leq \alpha \\ x + \alpha & x < -\alpha \end{cases} \]

An ST has been combined with a squeeze and excitation network with an attention mechanism to construct an adaptive residual shrinkage network for feature extraction in fault diagnosis by Zhao et al. [27]. In this work, we optimized the network structure to make it suitable for multivariate time series. We connected it with the MIC-based correlation layer introduced above to build a feature extraction module, as shown in the red section in Fig. 4.

The structure of the ST-based, adaptive, residual elimination subnetworks is shown in Fig. 5.

![Fig. 5. Adaptive residual elimination subnetwork structure.](image)

In this structure, the first step is to calculate the average value of all input data using global average pooling (GAP) and to restore it as the critical feature \( G \) in each channel. Along another path, the features after GAP in each channel are transferred to a small network consisting of a fully connected layer, a ReLU layer, and a final layer implementing the sigmoid function to acquire a coefficient \( k \), which is normalized to a value between 0 and 1. The final ST is \( G\times k \).

An advantage of this method is that the threshold \( \alpha \) can be automatically learned by training the network. Usually, such a threshold needs to be artificially set by engineers in advance. Moreover, the derivative of the output with respect to the input can only be either 0 or 1; in this sense, this derivative is similar to the ReLU activation function. Therefore, the risk of gradient vanishing and gradient explosion is also reduced in the deep networks.

### C. Step-ahead prediction module

Features filtered by residual shrinkage modules have lost their timing characteristics. To reutilize the redundant and temporal information between the input sequences and their own timing characteristics, the GRU layer was connected. The time-distributed dense layer was designed for full connection in the time dimension.
In addition, the Bi-GRU structure has been adopted to enhance sequential information [28], as shown in Fig. 3 in blue. Specifically, one GRU flows in the forward direction and calculates the forward hidden states ($\rightarrow h_1$, $\rightarrow h_2$, ..., $\rightarrow h_n$). Another GRU flows in the backward direction and calculates the backward hidden states ($\leftarrow h_1$, $\leftarrow h_2$, ..., $\leftarrow h_n$). The update of the hidden layer information at the current time in the Bi-GRU structure is related not only to $h_{t-1}$ but also to $h_{t+1}$, and the new hidden state is obtained by combining them, as follows:

$$h_t = h_{t-1} \oplus h_{t+1}$$  \hspace{1cm} (14)

### IV. CASE STUDY

With the development of the nuclear industry, monitoring technology based on the IIoT has been further applied in nuclear facilities to improve the reliability of system operation. The development of big data-based intelligent nuclear power technology cannot be separated from the support of the IIoT framework.

The monitoring process of nuclear system operations is mainly executed by the operators in the main control room (MCR). However, the long period of manual monitoring can introduce a tremendous mental burden to the operators. Additionally, the operators can only focus on the current situation in the system’s abnormal state. They cannot determine the evolution process to accurately predict and judge future situations. The famous historical Three Mile Island and Chernobyl nuclear accidents were both caused by operator overreliance on prior knowledge and improper estimation of follow-up developments, which introduced worsened conditions [29]. Therefore, it is essential to make short-term predictions of accidents to provide administrators with auxiliary decision information. This section presents a case study using a real IIoT platform in an advanced nuclear industrial process to illustrate the application and uses of the proposed multivariate time series prediction approach.

#### A. Data collection

The advanced China Lead-based Reactor (CLEAR) IIoT platform was proposed and optimized by the Institute of Nuclear Energy Safety Technology (INES) of Hefei Institute of Physical Science, Chinese Academy of Sciences (HFIPS, CAS) [22], [30], [31]. CLEAR has been regarded as the reference design for engineering projects for the development of accelerator-driven systems (ADSs) for nuclear waste transmutation since 2011. This platform combines a lead-bismuth experimental circuit, a nonnuclear test facility, and a full-scope simulator. The detailed structures and IIoT control center are shown in Fig. 6. Administrators can remotely monitor operating device status and achieve device positioning, remote transmission, and data modification in IIoT control centers.

To verify the effectiveness of the proposed method, we collect a large amount of experimental data from the above nuclear IIoT platform. Five types of time series in different operating scenarios are discussed. Most of these transients are common accidents in the coolant system, which is a crucial subsystem of the entire reactor. The scenario of coolant flow affects the speed of heat export and determines the reliability and economy of the overall operation of the reactor. The discussed scenario includes one minor accident in the core (20% accidental reactivity insertion), one serious accident scenario (loss-of-coolant accident (LOCA)), and three loss-of-flow rate accident scenarios in the primary circuit, secondary circuit, and hot sink. Information about each operating scenario is shown in Table I. A detailed description of the most serious LOCA can be found in reference 32.

![Fig. 6. Advanced nuclear platform (a): Designed structure of reactor; (b): Nonnuclear test facility (c): IIoT control center.](image)

![Fig. 7. Node division diagram of the designed coolant system.](image)

<table>
<thead>
<tr>
<th>Scenario No.</th>
<th>Description</th>
<th>Data size</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>20% power step</td>
<td>599<em>316</em>100</td>
</tr>
<tr>
<td>II</td>
<td>Loss-of-coolant accident (LOCA) [32]</td>
<td>484<em>316</em>100</td>
</tr>
<tr>
<td>III</td>
<td>Loss-of-heat-sink accident (LOHS)</td>
<td>477<em>316</em>100</td>
</tr>
<tr>
<td>IV</td>
<td>Loss-of-flow accident (LOFA)</td>
<td>424<em>316</em>100</td>
</tr>
<tr>
<td>V</td>
<td>Loss-of-feedwater accident (LOFW)</td>
<td>231<em>316</em>100</td>
</tr>
</tbody>
</table>

![Table I. ACCIDENT SCENARIO](image)
Fig. 7 shows the node division diagram of the designed system. There are 100 data samples for each scenario. Each sample is collected from 316 monitoring points corresponding to different components as follows: 800-Core temperature (K); core coolant temperature, flowrate (kg/m²), pressure (MPa) and control rod position (cm), reactivity (pcm) and power (MW); 370, 380, 470, 480-steam generator (SG) primary and secondary side temperature, flow rate and pressure; 535, 635, 735, 835-main pump temperature, flowrate, and pressure; 341, 441, 320, 420-fan primary and secondary side temperature, flowrate and pressure; 366, 466-feed pump temperature, flowrate, and pressure.

In Fig. 7, the initial number of each parameter represents the position of the relevant component in the node diagram. The length of each sample is varied according to the specific experimental conditions. The default sampling time of the designed monitoring system is 0.25 s. Therefore, a total of 500 samples provides 69,994,000 monitoring points.

To unify the data noise under different accident scenarios, the signal obtained from the experimental platform is processed by filtering. The different levels of the signal-to-noise ratio (SNR), following a Gaussian distribution, are then added artificially into the sampled dataset, which includes 20 dB, 10 dB, and 5 dB.

The definition of the SNR is shown as follows:

$$\text{SNR(dB)} = 20 \log_{10} \frac{A_n}{A_s}$$  \hspace{1cm} (15)

where $A_n$ and $A_s$ are the amplitude of the signal and noise, respectively.

We select the coolant temperature in the core inlet and the average SG secondary output temperature as the key prediction metrics in this paper. They determine the core’s operating power conditions and are strongly associated with the safety and reliability of the entire system. Additionally, the SG secondary output temperature is closely related to the quality of the load power output. Most research on control and monitoring strategies in nuclear industrial systems focuses on balancing these two metrics in nuclear industrial systems. All 500 samples were normalized by the max-min normalization method and divided into groups of 80% and 20% for training and testing, respectively.

**B. Initial hyperparameter setting**

Since there is no uniform standard for the setting of the hyperparameters, the initial values for model training were designed in accordance with the preliminary training results and previous experience. The basic principle was to reduce the complexity of the model and the training time as much as possible while maintaining prediction accuracy. Most of the hyperparameters were obtained from previous work, such as the filter and stride of the convolutional layer. Certain hyperparameters, such as the number of units in the Bi-GRU network time step, prediction step, and learning rate, are selected after a few preliminary experiments. The specific preliminary selections are shown in Table II.

The batch size significantly affects training. However, due to the limitations imposed by the length of the dataset, it is more beneficial to set the batch size to a smaller size.

**C. Learning rate optimization**

Under minibatch conditions, there will be an uncertain element in different batches to interfere with the model training, preventing the fixed learning rate from allowing the model to converge as quickly as possible.

Therefore, from the perspective of hyperparameter optimization, we focused on the model’s learning rate. The relationship between the rate decay ratio and the epoch is fixed in traditional methods. However, different models with large differences often do not achieve good results with the same formula. For better training, the popular learning rate decay (LRD) strategy, in which the learning rate is relatively high at the beginning of training, was adopted. However, as the training process continues, the learning rate is reduced in certain epochs. Losschilov’s research on decoupled weight decay regularization in ICLR 2019 also verified that LRD in adaptive moment estimation (Adam) optimization is a profitable operation in model training [33].

We divided the entire learning procedure (500 epochs) into ten intervals. The initial learning rate was 0.1, and the search range was 0.001/epoch to 0.1/epoch. The relationship between the initial learning rate and the learning rate in each interval is as follows:

$$\alpha = \frac{1}{1 + \text{Decayrate}*\text{epoch}} \alpha_0$$  \hspace{1cm} (16)

The decay rate in each interval is optimized one by one. The beginning observation of the next interval is the ending observation of the previous interval. The advantages of this approach are as follows:

1) The network will initially learn at a high rate and subsequently be restricted to a lower rate.
2) Different decay rates in different epochs can prevent the model from being trapped in a locally optimal solution.
3) The learning rate decays continuously during the entire training process to ensure that the network will not fall into overfitting.

Bayesian theory-based optimization (BTO) [34] was adopted to obtain the best rate of decay of the learning rate in each interval. This procedure has two essential components: the probabilistic agent model (PAM) and the acquisition function (AF). The PAM establishes the distribution that is

<table>
<thead>
<tr>
<th>Layer</th>
<th>Parameter</th>
<th>Designed value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCNN</td>
<td>Number</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Filter</td>
<td>128→32→64</td>
</tr>
<tr>
<td>Bi-GRU</td>
<td>Units</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Time step</td>
<td>6</td>
</tr>
<tr>
<td>Others</td>
<td>Batch size</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Learning rate</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Loss function</td>
<td>Root-mean-square error (RMSE)</td>
</tr>
<tr>
<td></td>
<td>Optimizer</td>
<td>Adam</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Layer</th>
<th>Parameter</th>
<th>Designed value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCNN</td>
<td>Number</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Filter</td>
<td>128→32→64</td>
</tr>
<tr>
<td>Bi-GRU</td>
<td>Units</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Time step</td>
<td>6</td>
</tr>
<tr>
<td>Others</td>
<td>Batch size</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Learning rate</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Loss function</td>
<td>Root-mean-square error (RMSE)</td>
</tr>
<tr>
<td></td>
<td>Optimizer</td>
<td>Adam</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Layer</th>
<th>Parameter</th>
<th>Designed value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCNN</td>
<td>Number</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Filter</td>
<td>128→32→64</td>
</tr>
<tr>
<td>Bi-GRU</td>
<td>Units</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Time step</td>
<td>6</td>
</tr>
<tr>
<td>Others</td>
<td>Batch size</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Learning rate</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Loss function</td>
<td>Root-mean-square error (RMSE)</td>
</tr>
<tr>
<td></td>
<td>Optimizer</td>
<td>Adam</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Layer</th>
<th>Parameter</th>
<th>Designed value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCNN</td>
<td>Number</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Filter</td>
<td>128→32→64</td>
</tr>
<tr>
<td>Bi-GRU</td>
<td>Units</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Time step</td>
<td>6</td>
</tr>
<tr>
<td>Others</td>
<td>Batch size</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Learning rate</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Loss function</td>
<td>Root-mean-square error (RMSE)</td>
</tr>
<tr>
<td></td>
<td>Optimizer</td>
<td>Adam</td>
</tr>
</tbody>
</table>
assumed to be obeyed by the input data. The AF is used to select the following objective observations.

In each iteration, the maximum AF is confirmed based on the previously mentioned PAM conditions. This approach often leads to a tradeoff between the distribution of the observations and the improvement of $f(x)$. This procedure is referred to as the exploration-exploitation tradeoff. The new observations are passed as input to the system, and new outputs are obtained to update $D$ and the PAM.

In this paper, we propose an optimized BTO (OBTO) using the Thompson sampling (TS) [35] technique as the AF. We then combine the Latin hypercube design (LHD) [36] to initialize the search space. Compared with the traditional AF, such as the upper confidence bound (UCB) and expected improvement (EI), advantages of this design method include the following:

1) The distribution of the sample space becomes more even according to the probability density to reduce the search costs.
2) The exploration is executed in the area of large variance, which can obtain a higher optimization target.
3) The exploitation becomes more precise based on the posterior distribution of the PAM.

<table>
<thead>
<tr>
<th>Operating scenario</th>
<th>DBN-Bi-GRU</th>
<th>CNN-Bi-GRU</th>
<th>2-Conv-Bi-GRU Autoencoder</th>
<th>DCGNet with Bi-GRU</th>
<th>VMD-Bi-GRU</th>
<th>EWT-NCUBi-GRU</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core inlet temperature</td>
<td>MAPE (%)</td>
<td>MAPE (%)</td>
<td>MAPE (%)</td>
<td>MAPE (%)</td>
<td>MAPE (%)</td>
<td>MAPE (%)</td>
<td>MAPE (%)</td>
</tr>
<tr>
<td>I</td>
<td>II</td>
<td>III</td>
<td>IV</td>
<td>V</td>
<td>I</td>
<td>II</td>
<td>III</td>
</tr>
</tbody>
</table>
The optimization procedure is shown in **Algorithm 1**.

**Algorithm 1:** Improved BO-based search procedure

**Input:** Input data: $X$; target function between the rate of decay of the learning rate and the loss: TS based acquisition function: $f_\epsilon$; probabilistic agent model: $PAM$

$D=\{(x_i,f(x_i))\} (x_i,f(x_i)), \ldots, (x_i,f(x_i)) \}$ Initial by LHD

\[ p(y \mid x,D) = (PM,D) ^* \text{Assume that the input data follow the distribution of PAM} \]

1: for $i=1,2, \ldots, T$ do
  2: $x_i = \arg \max_{x \in X} f_\epsilon(x,D_{i-1})=\arg \max_{x \in X} f_\epsilon(x,p(y \mid x,D_{i-1}))$
  3: $y_i = f(x_i) + \epsilon_i$ * Evaluate the objective function value
  4: $D_i = D_{i-1} \cup \{x_i, y_i\}$ end for * Assess and update PAM

5: end for

**V. RESULTS AND DISCUSSION**

**A. Benchmark models**

We compare the prediction performance against six state-of-the-art, DL-based benchmark models, which contain two DL models with traditional signal processing and four hybrid networks, as follows:

1) **DBN-GRU:** This model [15] combines the three-layer-based deep Bayesian network (DBN) with the GRU network.

2) **CNN-LSTM:** This model [16] contains two traditional convolutional layers, one LSTM layer, and two dense layers.

3) **2D-ConvLSTM Autoencoder:** This model [18] uses the two convolutional LSTM (convLSTM) models as the encoding layers and the two Bi-LSTM models as the decoding layers to complete the entire prediction model.

4) **DCGNet:** The deep convolutional neural network and gated recurrent unit network (DCGNet) model [19] uses PCA to select the input feature. The model is then connected to the two feature extraction modules, which is similar to our previous work related to fault diagnosis [30]. Multivariate coupling feature extraction is performed with the 2-D CNN, 1-DCNN, GRU, and fully connected layer. The dynamic feature extraction only contains the parallel GRU. The final output is weighted and connected with the above two channels.

5) **VMD-GRU:** VMD is employed in this model [20] to decompose complex data into simplified modes. The GRU-based prediction model, comprising a GRU input layer, multiple fully connected layers, and a rank-ordered terminal layer, is then trained for each mode to produce prediction results.

6) **EWT-NCULSTM:** This model [21] uses the EWT to decompose the input signal into high- and low-frequency time series features. These features are connected to the designed new cell update LSTM (NCULSTM) to make a further prediction.

**B. Evaluation index**

Two important indexes, namely, the mean absolute percentage error (MAPE) and coefficient of determination ($R^2$), are employed as the evaluation indexes as follows:

$$MAPE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i| \times 100}{n} \quad (17)$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \quad (18)$$

The MAPE is the sum of each absolute error divided by the actual value and is the average percentage error. The $R^2$ reflects the amount of variation of the dependent variable that is explained by the independent variable. The closer its value is to 1, the better the prediction effect is.

**C. Implementation Environment**

The training procedure of the proposed DL-based prediction model is implemented based on an Intel Core i7-10750 2.6 GHz central processing unit (CPU) with 64 GB RAM and accelerated by the Compute Unified Device Architecture (CUDA) v11.1 environment with an NVIDIA RTX 3070 graphics processing unit (GPU). The development was performed using Python 3.8.11 and Keras v2.2.4 with TensorFlow 1.1.4.0.

**D. Comparisons with benchmark models**

In this section, we put Bi-GRU into each benchmark model in Section V part A to unify the structure of the models to be compared. In addition, the format of the decomposed subset of the model using VMD and EMT operations is unified to 3. The purpose of these operations is to make fairer comparisons with the benchmark model, highlighting the advantages of the designed architecture.

**Fig. 8.** Comparison of training time ((a) Core inlet temperature; (b) SG outlet temperature).
2) VMD-Bi-GRU and EMT-NCUBi-GRU with a noise elimination structure and the proposed method exhibited satisfactory predictive performance under a 20 dB-noise environment. It is worth noting that the DCGNet with Bi-GRU using PCA also achieves good results in most cases and has certain advantages over other models in terms of training complexity. This result indicates that this combination has a specific value in a weak data uncertainty environment.

3) With the increase in data uncertainty, ordinary hybrid neural networks such as CNN-Bi-GRU, 2D ConvBi-GRU Autoencoder, and DCGNet with Bi-GRU do not perform well for prediction tasks. Furthermore, the proposed method obtained ideal results in most operating scenarios. In particular, under the 5 dB-noise environment, the prediction index $R^2$ was increased by 16.26% and 15.04% on average. MAPE was decreased by 24.49% and 24.52% on average, highlighting the advantages of the designed structure in extracting residual features.

4) The average training time under a single epoch of the proposed method was only 26.77% and 27.95% compared with VMD-Bi-GRU and EMT-NCUBi-GRU, respectively. This huge gap is observed because the algorithm complexity of VMD and EMT is proportional to the number of decomposed modes or the number of subsets obtained. Although it can be solved by parallel computing to reduce the training time, the computing power resources occupied will not be reduced. The designed network adopts the adaptive threshold to achieve noise elimination. The advantages of the end-to-end structure of the deep network are retained.

E. Comparisons of different methods for LRD optimization

In this section, we compare the optimization of the designed model using random search, particle swarm optimization (PSO) [37], enhanced gray wolf optimization (EGWO) [5], traditional BTO [38], and the proposed OBTO strategy.

Similarly, the initial search space of the learning rate and overall number of iterations are unified to fairly compare the results. Table IV shows the optimization results of different methods under different noise levels with ten iterations. The average optimization time with each method is also included. The following results are obtained:

1) The random search strategy has an advantage in terms of time complexity, but its performance varies considerably in each optimization run.

2) PSO and EGWO achieve good optimization results. EGWO has strong competitiveness in both prediction performance improvement and stability. However, the time required for each optimization run is overly long, which is verified in the comparison result of reference 5.

3) Although the BTO improvement effect is not as good as that of PSO, the time required to achieve a similar performance target is significantly less (only 53.02% of the time required for PSO on average).

4) Although the proposed OBTO cannot improve the index compared with the BTO, it significantly reduces the optimization time by 30.04%. Moreover, compared with the results from 10 iterations, good results are already obtained from 6 iterations in a relatively short time, illustrating the superiority of the proposed BTO-based method.

<table>
<thead>
<tr>
<th>Noise level</th>
<th>Decrease of MAPE</th>
<th>Increase of $R^2$</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random search</td>
<td>$\pm 0.021 \pm 0.037 \pm 0.102$</td>
<td>$\pm 0.571 \pm 1.632 \pm 2.775$</td>
<td>18.13</td>
</tr>
<tr>
<td>PSO</td>
<td>$\pm 0.004 \pm 0.016 \pm 0.032$</td>
<td>$\pm 0.089 \pm 0.116 \pm 0.184$</td>
<td>135.45</td>
</tr>
<tr>
<td>EGWO</td>
<td>$\pm 0.005 \pm 0.009 \pm 0.023$</td>
<td>$\pm 0.061 \pm 0.068 \pm 0.146$</td>
<td>116.24</td>
</tr>
<tr>
<td>Traditional BTO</td>
<td>$\pm 0.008 \pm 0.022 \pm 0.041$</td>
<td>$\pm 0.113 \pm 0.235 \pm 0.377$</td>
<td>71.86</td>
</tr>
<tr>
<td>OBTO</td>
<td>$\pm 0.016 \pm 0.028$</td>
<td>$\pm 0.033 \pm 0.067 \pm 0.148$</td>
<td>50.28</td>
</tr>
<tr>
<td>OBTO with 6 iterations</td>
<td>$\pm 0.009 \pm 0.012 \pm 0.028$</td>
<td>$\pm 0.044 \pm 0.091 \pm 0.200$</td>
<td>32.61</td>
</tr>
</tbody>
</table>

F. Final prediction results of the test dataset

Fig. 9 further shows the final prediction results of the proposed deep hybrid model optimized by OBTO for the 5-dB test datasets with the following results:

1) The 95% confidence intervals of the prediction models designed for different operating scenarios contain the actual theoretical output results and follow the increasing trend of prediction metrics for CLEAR under high-level noise environments.

2) Compared with the actual value, the fluctuation in the predicted results from the designed model is less. The mean value is similar to the theoretical value, which indicates that the proposed network can effectively eliminate the data uncertainty and achieve good performance for the step-ahead prediction.

3) The actual lengths of time taken to make a step-ahead prediction for each operating scenario (model test time) are 0.64, 0.35, 0.35, 0.32, and 0.28 s for the core inlet temperature and 0.69, 0.38, 0.37, 0.33, and 0.29 s for the average SG outlet temperature, achieving the requirement for transient prediction in advance.

VI. Conclusions

This paper focuses on developing an end-to-end, deep hybrid network-based, multivariate short-term time-series prediction framework in the IoT environment-based industrial process. The MIC is adopted to build the variate selection module. The CNN with a residual shrinkage module is built to reduce the training time, the computing to time complexity, but its performance varies considerably in each optimization run.

The random search strategy has an advantage in terms of time complexity, but its performance varies considerably in each optimization run.

2) PSO and EGWO achieve good optimization results. EGWO has strong competitiveness in both prediction performance improvement and stability. However, the time required for each optimization run is overly long, which is verified in the comparison result of reference 5.

3) Although the BTO improvement effect is not as good as that of PSO, the time required to achieve a similar performance target is significantly less (only 53.02% of the time required for PSO on average).

4) Although the proposed OBTO cannot improve the index compared with the BTO, it significantly reduces the optimization time by 30.04%. Moreover, compared with the results from 10 iterations, good results are already obtained from 6 iterations in a relatively short time, illustrating the superiority of the proposed BTO-based method.
performance of the developed network under different noise environments has been validated through comparisons with other state-of-the-art methods in a case study based on the established CLEAR industrial platform. Furthermore, we will incorporate the designed deep hybrid model into the proposed prognostic and health management (PHM)-based auxiliary decision framework [39].

The proposed method can also be applied to the many-to-many time series prediction tasks of other IIoT environments.

In the future, we will also seek cooperative endeavors to solve specific prediction problems and verify the effectiveness of our proposed method in different industrial areas.

In future work, we will combine the proposed method to further optimize the periodic power prediction task on a medium time scale and the remaining useful life prediction (RUL) task on a large time scale. In particular, research related to DL-based prediction model uncertainty (epistemic uncertainty) is an avenue for future study [40].

![Image](image_url)  
**Fig. 9.** Prediction results for 5 dB test datasets ((a) Core inlet temperature; (b) SG outlet temperature).

## References


