LETTER

A Novel Frequency Hopping Prediction Model Based on TCN-GRU*

Chen ZHONG[†], **Chegnyu WU**[††], **Xiangyang LI**[†††], **Ao ZHAN**[††a)], ***Nonmembers***, ***and*** **Zhengqiang WANG**[††††], ***Member***

SUMMARY A novel temporal convolution network-gated recurrent unit (NTCN-GRU) algorithm is proposed for the greatest of constant false alarm rate (GO-CFAR) frequency hopping (FH) prediction, integrating GRU and Bayesian optimization (BO). GRU efficiently captures the semantic associations among long FH sequences, and mitigates the phenomenon of gradient vanishing or explosion. BO improves extracting data features by optimizing hyperparameters besides. Simulations demonstrate that the proposed algorithm effectively reduces the loss in the training process, greatly improves the FH prediction effect, and outperforms the existing FH sequence prediction model. The model runtime is also reduced by three-quarters compared with others FH sequence prediction models.

key words: *frequency hopping prediction, constant false alarm rate, temporal convolution network-gated recurrent unit, Bayesian optimization*

1. Introduction

Frequency-hopping (FH) communication is widely used in common anti-jamming communications due to its strong anti-interference ability, ease of multi-access networking, and superior security [1]. Accurate tracking of interference is achieved by identifying and distinguishing different FH code sequences, intelligently capturing FH sequences, projecting the FH center frequency at future moments [2]. How to improve FH sequence prediction is thus a hot topic in academia and industry recently.

With the continuous development of deep neural network, a long short-term memory (LSTM) model is proposed to realize high-frequency sequence prediction, which predicts the frequency point by point based on historical high-frequency frequency [3]. In [4], they perform phase space reconstruction of FH sequences generated by three chaotic maps, and then establish a TCN-based sequence prediction model to realize FH sequence prediction in phase space. However, the above works are researched under ideal environmental conditions.

Terrain, wind speed, humidity and other factors deteriorate the accuracy of FH signal detection in realistic scenarios. Constant false alarm rate (CFAR) detection algorithm is proposed to detect FH for eliminating Non-stationary and non-uniform complex clutter [5]. Deng et al. [6] employ CFAR algorithm, and propose TCN modeling and BO for predict FH sequences, which has noise tolerance, stable gradients and faster training speed. The greatest of constant false alarm rate detection (GO-CFAR) algorithm is proposed as a well detecting mechanism for extracting targets in clutter and noise [7]. The algorithm achieves good detection performance in both clutter edges and uniform clutter environments. In this letter, we propose a novel temporal convolution network-gated recurrent unit (NTCN-GRU) model, employing the TCN model, integrating a gated recurrent unit (GRU) for a based GO-CFAR FH system. In the proposed NTCN-GRU, the GRU model learns the long-term dependency and temporal dynamics of the sequence, and Bayesian optimization (BO) is used to optimize the model hyperparameters to improve the prediction accuracy. Moreover, self-attention mechanism captures the internal correlation of FH data or features. Simulation results show that the proposed algorithm model improves the FH prediction and achieve large performance gain over other algorithms with lower computational complexity.

2. System Model

The total FH sequence is assumed to be $x_{1:N}$ with length N, including input and predicted FH sequence, shown in Fig. 1. The system model is denoted as:

$$\hat{x}_{(2i-1)T+1:2iT} = \mathrm{Pred}(x_{2(i-1)T+1:(2i-1)T}) \quad i \in (1, 2, \ldots, \lfloor N/2T \rfloor) \tag{1}$$

where Pred$(\cdot)$ is the FH prediction method.

In Fig. 1, the predicted FH sequences in duration T are

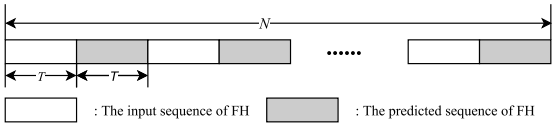

Fig. 1 System model.

Manuscript received October 23, 2023.
Manuscript revised March 17, 2024.
Manuscript publicized April 19, 2024.

[†]School of Computer Science and Technology, Zhejiang Sci-Tech University, Hangzhou, 310018, P.R.China.
[††]School of Information Science and Engineering, Zhejiang Sci-Tech University, Hangzhou, 310018, P.R.China.
[†††]Southwest China Institute of Electronic Technology, Chengdu, 610036, P.R.China.
[††††]School of Communication and Information Engineering, Chongqing University of Posts and Telecommunication, Chongqing, 400065, P.R.China.

*This work was supported by the Open Fund of Key Laboratory of Big Data Intelligent Computing, Chongqing University of Posts and Telecommunications under grant BDIC-2023-B-002.

a) E-mail: zhanao1983@zstu.edu.cn (Corresponding author)

DOI: 10.1587/transfun.2023EAL2095

predicted per input FH sequence of duration T, and then $\lfloor N/2T \rfloor$ durations of pridicted FH are obtained.

3. Proposed NTCN-GRU Model

As shown in Fig. 2, NTCN-GRU is proposed for GO-CFAR detection, which includes TCN, GRU, BO and self-attention mechanism. The BO optimizes the hyperparameters of the TCN-GRU model, i.e., kernel size, number of kernels, dropout factor and number of stacks, the optimized hyperparameters are organized into set $\boldsymbol{s}$, $\boldsymbol{S}$ is the hyperparameter search space, $\boldsymbol{s} \epsilon \boldsymbol{S}$. The self-attention mechanism captures the internal correlation of data or features, reducing the dependence on external information.

3.1 TCN

TCN uses a 1-D fully convolution network and padding to ensure that the duration of each hidden layer is equal to that of the input layer. Causal convolution can ensure that future information does not leak from the past [8]. The sequence model is requested to meet two principles, i.e., 1) the output duration of the model is equal to the input duration; 2) when processing the current time step, the leaked information of the future time step cannot be known.

The input sequence $x_{1:T}$ includes FH data from moment 1 to T, where T is assumed to be the length of the input FH. Without loss of generality, we also assume the length of the predicted FH is T.

As shown in Fig. 4, in order to realize Eq. (1) in this paper, the T TCN models are run to obtain the predicted sequence $\hat{x}_{T+1:2T}$.

For simplicity, we show the TCN process of getting the prediction sequence $\hat{x}_{T+1:2T}$ by using input sequence $x_{1:T}$ in Fig. 4, $\hat{x}_{T+1:2T}$ and $e_{k-1} = \hat{x}_{k:T} - x_{k:T}$ are employed to input the next TCN for more accurate prediction. The prediction by using TCN are expressed as

$$\begin{cases} (e_1, \hat{x}_{T+1}) = TCN(x_{1:T}, e_0) & k = 1 \\ (e_k, \hat{x}_{T+k}) = TCN(x_{k:T}, \hat{x}_{T+1:T+k-1}, e_{k-1}) & k \in [2, T] \end{cases} \tag{2}$$

where $e_0 = 0$.

3.2 GRU

Gated recurrent unit (GRU) is a commonly used gated Recurrent Neural Network (GRNN) that controls the information flow by means of learnable gates. The GRU learns the long term dependencies and temporal dynamics of the FH sequences, which can provide information about the frequency changes. Meanwhile, GRU has fewer parameters than LSTM, which usually requires less computation during training [9].

For simplicity, we define $x_{i:i}$ as x_i. The input of GRU is the FH data x_i and h_{i-1}, and the output is state h_i, which is denoted as

$$\begin{cases} h_i = GRU(x_i, h_{i-1}) & i \in (1, 2, \ldots, T) \\ h_i = GRU(\hat{x}_i, h_{i-1}) & i \in (T+1, T+2, \ldots, 2T) \end{cases} \tag{3}$$

Figure 5 shows the GRU processing for input sequence $x_{1:T}$ and predicted sequence $\hat{x}_{T+1:2T}$.

GRU consists of a reset gate R_i and an update gate Z_i, which is used to capture short-term dependencies in FH

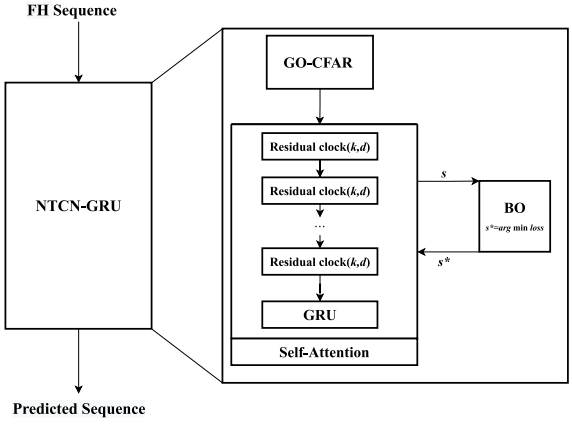

Fig. 2 Model diagram of NTCN-GRU.

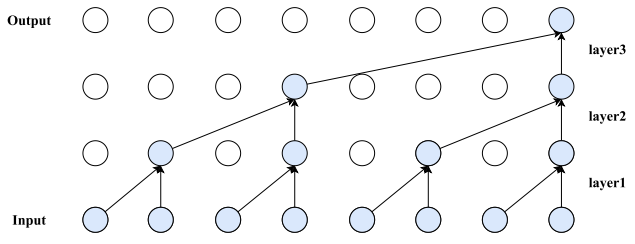

Fig. 3 1D causal convolution.

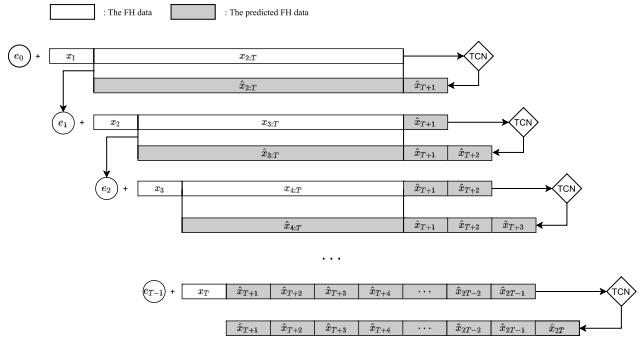

Fig. 4 TCN prediction process.

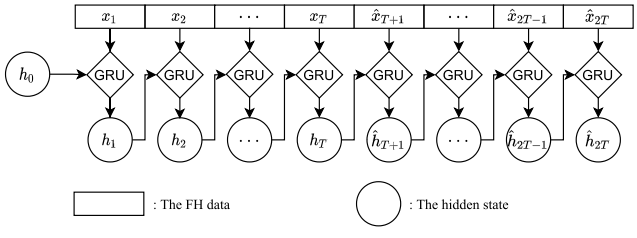

Fig. 5 The GRU processing for $x_{1:T}$ and $\hat{x}_{T+1:2T}$.

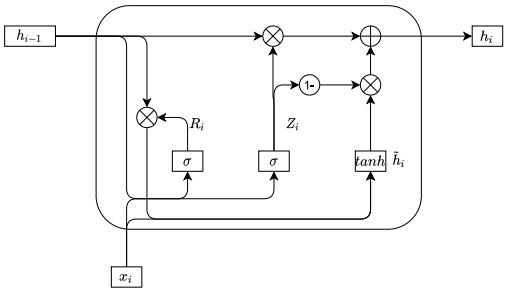

Fig. 6 GRU model.

sequences and capture long-term dependencies in FH sequences respectively. R_i and Z_i are shown as follows

$$\begin{aligned} R_i &= \sigma\Big(W_r \cdot [h_{i-1:i-1}, x_i]\Big) \\ Z_i &= \sigma\Big(W_z \cdot [h_{i-1:i-1}, x_i]\Big) \end{aligned} \tag{4}$$

σ is a sigmoid function that changes the data to a value in the range [0, 1]. Linear transformation of the matrix formed by splicing x_i and h_{i-1} by the weight matrix W_r and W_z.

As shown in Fig. 6, $\widetilde{h_i}$ is a candidate hidden state, which is used to assist the hidden state calculation afterwards. The candidate hidden state for time step i is

$$\widetilde{h_i} = \tanh\Big(W_h \cdot [R_i * h_{i-1}, x_i]\Big) \tag{5}$$

where W_h is the corresponding weight matrix and *tanh* changes the data to values in the range $[-1, 1]$.

h_i of the current time step can be obtained after the above calculations.

$$h_i = (1 - Z_i) \odot h_{i-1} + Z_i \odot \widetilde{h_i} \tag{6}$$

The $\odot$ operation multiplies the corresponding elements in the matrix.

3.3 Bayesian Optimization

Then set $\boldsymbol{s}$ is input into the model for prediction. The computed *loss* is finally returned to the BO module for iterative optimization. The optimization is formulated as

$$\boldsymbol{s}^* = \arg\min loss \tag{7}$$

$$loss = \text{MSE}(e) \tag{8}$$

where Pred_s is the predicted sequence of the corresponding set of hyperparameters. The BO process is shown as Algorithm 1.

Assume that the agent model for Bayesian hyperparameter optimization is *GP* (Gaussian Process) [10]. Enter the set of hyperparameter intervals $\boldsymbol{S}$ as well as *GP*.Get the initialization dataset D by InitSample(·). By randomly selecting the hyperparameter values in $\boldsymbol{S}$ several times, and evaluating each set of hyperparameters with *Pred*, we get the corresponding evaluation result *loss*, deposit the combination of $(\boldsymbol{s}, loss)$ into D. The initial dataset D is given by

Algorithm 1 BO

```
Input:S,GP
D←InitSample(Pred, S)
for i ← |D| do
    p(loss_s|s, D) ← FitModel(GP, D)
    s* ← arg max_{s∈S} EI(s, p(loss_s|s, D))
    loss_{s*} ← Pred(s*)
    D ← D ∪ (s*, loss_{s*})
end for
```

the following equation

$$D = \Big\{(\boldsymbol{s}_1, loss_{s_1}), (\boldsymbol{s}_2, loss_{s_2}), \dots\Big\} \tag{9}$$

Assume the *loss* obeys a Gaussian distribution [11]

$$loss_s \sim N(\mu_s, \sigma_s^2) \tag{10}$$

In (10), we obtained the mean μ_s and variance σ_s^2 of the loss by the following two equations

$$\mu_s = k(\boldsymbol{s})^T(\mathbb{k} + \sigma_N^2 I)^{-1} loss_s \tag{11}$$

$$\sigma^2 = \mathbb{k}(\boldsymbol{s}, \boldsymbol{s}) - k(\boldsymbol{s})^T(\mathbb{k} + \sigma_N^2 I)^{-1} k(\boldsymbol{s}) \tag{12}$$

where I is the unit matrix, $\mathbb{k}$ is the covariance function, and $k(\boldsymbol{s})$ is the Euclidean distance between the set of hyperparameters generated.

For reducing the amount of computational data, the number of prescribed iterations for a given BO is calculated with the following

$$p(loss_s|\boldsymbol{s}, D) \leftarrow FitModel(GP, D) \tag{13}$$

The acquisition function is employed to equalize the ratio between the mean and the variance to find the optimal value of this hyperparameter. Capturing the ratio between mean and variance by EI, which is given by the formula

$$\begin{aligned} EI(\boldsymbol{s}) = &\Big(loss_{\min} - \mu(\boldsymbol{s})\Big)\Phi\left(\frac{loss_{\min} - \mu(\boldsymbol{s})}{\sigma(\boldsymbol{s})}\right) \\ &+ \sigma(\boldsymbol{s})\phi\left(\frac{loss_{\min} - \mu(\boldsymbol{s})}{\sigma(\boldsymbol{s})}\right) \end{aligned} \tag{14}$$

$$\mu(\boldsymbol{s}) = \max\Big(0, loss_{\min} - loss_s\Big) \tag{15}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the cumulative density (CDF) and probability density (PDF) of the normal distribution.

Update the dataset D by depositing the trained hyperparameter combinations and corresponding metrics into the dataset D.

3.4 Self-Attention Mechanism

Based on the self-attention mechanism, we achieve matrix parallel computation using CUDA matrix multiplication, which can better utilize the hardware resources and speed up the training of the model. The input matrix is divided into multiple small matrices and assigned to different threads for

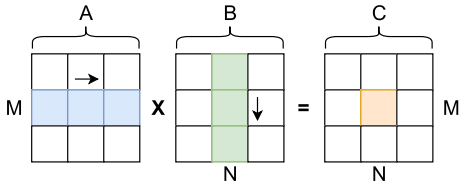

Fig. 7 CUDA matrix multiplication.

parallel computation.

The input FH sequence is linearly transformed to obtain Q (query), K (key) and V (value) matrices, the formula is as follows

$$Attention(Q, K, V) = Soft\max\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{16}$$

where d_k is the dimension size of K.

4. Simulation

The FH prediction simulation experiments were conducted with servers equipped with 64 GB of RAM, AMD Ryzen 95950X 16-core processors, GeForce RTX 4060Ti and GeForce RTX 3060. The core of FH technology is to generate FH sequence that meet the requirements, Logistic chaotic mapping is the simplest and most effective chaotic system [12], which is widely used in most chaotic encryption algorithms. In this letter, we generate pseudo-random FH sequence based on Logistic mapping, defined as

$$f(x_i) = r \cdot x_{i-1} \cdot (1 - x_{i-1}) \tag{17}$$

where x_i is the FH sequence value at moment i, and r is a control parameter, when $3.5699456 < \mu \leq 4$, the Logistic mapping has chaotic nature.

We use logistic mapping to generate FH sequence of duration 5000, of which 80% is used as a training set and 20% as a test set.

Data analysis is affected by the fact that different assessment indicators have various scales and units of measurement. In order to eliminate the effect of scales between indicators, it is necessary to standardize the data, i.e., normalize the data to between $[0, 1]$. The Z-value standardization method is adopted to scale the data. The standardization formula is as follows

$$x' = \frac{x - \mu}{\sigma} \tag{18}$$

The root mean square error (RMSE) between the predicted sequence and the original sequence is taken as a criterion for prediction performance as below

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N_{test}}(TCN_{x_{i-1}} - x_i)^2} \tag{19}$$

The optimal set of hyperparameters s^* obtained by BO update optimization is input into the test set, and the mean square error (MSE) is calculated to evaluate the model performance, s^* as shown in the Table 1.

Table 1 Model optimal hyperparameters.

Hyperparameters	Value
Kernel size	12
Number of kernels	37
Dropout factor	0.0411315
Number of stacks	2

Table 2 Running time of different models.

Model	Runtime(s)
GRU	75.607924
TCN	32.321640
TCN+GRU	50.698445
TCN+BO	5051.98436
NTCN-GRU	**1459.65561**

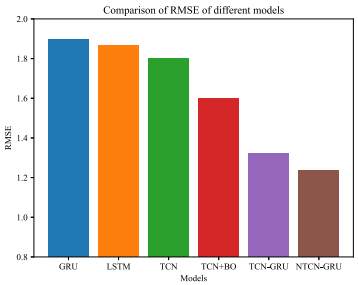

Fig. 8 Comparison of RMSE of different models.

Due to the fact that BO requires constant updating iterations, the time cost required is high. In order to solve this problem, we adopt the GRU structure. GRU is an improvement of LSTM, which not only simplifies the structure, but also improves the computational efficiency and better prediction, solves the problem of local optimization to a certain extent. As shown in Table 2 below, the running time of the model rises significantly after adopting BO, while the GRU structure can reduces the running time and improve the prediction efficiency.

Figure 8 shows the RMSE comparison results of the model NTCN-GRU with other FH sequence prediction models. With the same dataset, both the GRU and BO have an effect on FH sequence prediction. The NTCN-GRU model training is the best, GRU can reduce the model training time while ensuring the training efficiency and effectiveness.

As shown in Fig. 9, the actual target sequence is the red solid line. The TCN+GRU model with orange dots predicts the overall trend of the FH sequence more accurately, but the prediction accuracy is not high; the TCN+BO shown in dark blue outperforms the TCN+GRU, but the overall accuracy is still not high and the error is large; the NTCN-GRU model shown in light blue is the most effective, and the overall trend is basically overlapped with the trend of the actual FH sequence, and the accuracy is significantly higher than the rest of the comparison models.

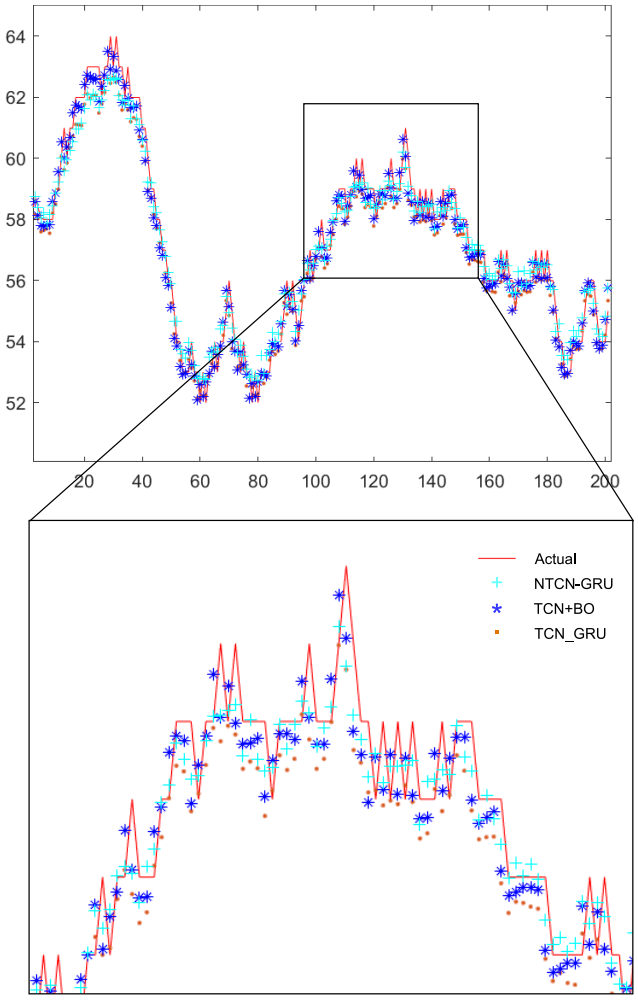

Fig. 9 Comparison of different model predictions.

5. Conclusion

In this letter, a NTCN-GRU modeling algorithm is proposed for the FH sequence prediction problem. First, the FH sequences are detected by the GO-CFAR algorithm to reduce the impact of interfering factors on FH prediction. Then, some hyperparameters of the prediction model are optimized with BO. The self-attention mechanism is employed for solves the long-distance dependence problem by calculating the interactions between hopping frequencies. Simulation results show that NTCN-GRU improves the prediction accuracy compared to the comparison model, and reduces the loss in the training process effectively as well.

References

[1] J. Zhu, A. Wang, W. Wu, Z. Zhao, Y. Xu, R. Lei, and K. Yue, "Deep-learning-based recovery of frequency-hopping sequences for anti-jamming applications," Electronics, vol.12, no.3, p.496, 2023.
[2] H. Pang, "Research on prediction algorithm of frequency hopping code sequence," Master thesis, University of Electronic Science and Technology of China, School of Aeronautics and Astronautics, 2021.
[3] G. Li, J. Xu, W. Shen, W. Wang, Z. Liu, and G. Ding, "Lstm-based frequency hopping sequence prediction," 2020 International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, pp.472–477, IEEE, 2020.
[4] G. Li, W. Wang, and G. Ding, "Chaotic frequency hopping prediction based on temporal convolutional network," 13th International Conference on Wireless Communications and Signal Processing, WCSP 2021, Changsha, China, pp.1–5, IEEE, 2021.
[5] K. Liu, Y. Li, P. Wang, X. Peng, H. Liao, and W. Li, "A CFAR detection algorithm based on clutter knowledge for cognitive radar," IEICE Trans. Fundamentals, vol.E106-A, no.3, pp.590–599, March 2023.
[6] Z. Deng, K. Lai, and J. Lei, "A temporal convolutional network based on bayesian optimization for frequency hopping prediction," 14th IEEE International Conference on Advanced Infocomm Technology, ICAIT 2022, Chongqing, China, pp.13–18, IEEE, 2022.
[7] J. Akhtar and K.E. Olsen, "GO-CFAR trained neural network target detectors," 2019 IEEE Radar Conference, Boston, MA, United states, pp.1–5, IEEE, 2019.
[8] W. Sheng, K. Liu, D. Jia, S. Chen, and R. Lin, "Short-term load forecasting algorithm based on LST-TCN in power distribution network," Energies, vol.15, no.15, pp.1–13, 2022.
[9] H.T. Guo, L. Pan, J. Wang, X.B. Fan, J. Li, and Z. Liu, "Short-term wind power prediction method based on TCN-GRU combined model," 2021 IEEE Sustainable Power and Energy Conference (iSPEC), Nanjing, China, pp.3764–3769, 2021.
[10] B. Shahriari, K. Swersky, Z. Wang, R.P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of Bayesian optimization," Proc. IEEE, vol.104, no.1, pp.148–175, 2016.
[11] J. Snoek, H. Larochelle, and R.P. Adams, "Practical Bayesian optimization of machine learning algorithms," Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems, Lake Tahoe, Nevada, United States, pp.2960–2968, Dec. 2012.
[12] Z. Chen, D. Liang, X. Deng, and Y. Zhang, "Performance analysis and improvement of logistic chaotic mapping," Journal of Electronics & Information Technology, vol.38, no.6, pp.1547–1551, 2016.