Multivariate Resource Usage Prediction With Frequency-Enhanced and Attention-Assisted Transformer in Cloud Computing Systems

Jing Bi, *Senior Member, IEEE*, Haisen Ma, *Student Member, IEEE*, Haitao Yuan, *Senior Member, IEEE*, Rajkumar Buyya, *Fellow, IEEE*, Jinhong Yang, Jia Zhang, *Senior Member, IEEE*, and MengChu Zhou, *Fellow, IEEE*

***Abstract*—Resource usage prediction in cloud data centers is critically important. It can improve providers' service quality and avoid resource wastage and insufficiency. However, the time series of resource usage in cloud environments is characterized by multidimensional, nonlinear, and high-volatility characteristics. Achieving high-accuracy prediction for time series with such characteristics is necessary but difficult. Traditional prediction methods based on regression algorithms and recurrent neural networks cannot effectively extract nonlinear features from data sets. Besides, many deep learning models suffer from gradient explosion or gradient vanishing during the training stage. Current commonly used prediction methods fail to uncover some vital information about the frequency domain features in the time series. To resolve these challenges, we design a Forecasting method based on the Integration of a Savitzky–Golay (SG) filter, a frequency enhanced decomposed transformer (FEDformer) model, and a frequency-enhanced channel attention mechanism (FECAM), named FISFA. It adopts the SG filter to reduce noise and smooth sequences in the raw sequences of resources. Then, we develop a hybrid transformer-based model integrating FEDformer and the FECAM, effectively capturing the frequency domain patterns. Besides, a meta-heuristic optimization algorithm, i.e., genetic simulated annealing-based particle swarm optimizer, is proposed to optimize key hyperparameters of FISFA. Then, FISFA predicts the future needs for multidimensional resources in highly fluctuating traces in real-life cloud environments. Experimental results demonstrate that FISFA achieves higher accuracy and performs more efficient prediction than several benchmark forecasting methods with realistic data sets collected from Alibaba and Google cluster traces. FISFA improves the prediction accuracy on average by 32.14%, 25.49%, and 27.71% over vanilla long short-term memory, transformer, and Informer methods, respectively.**

***Index Terms*—Cloud computing, deep learning, frequency enhancement, Savitzky–Golay (SG) filter, time series prediction.**

Manuscript received 8 February 2024; revised 11 April 2024; accepted 28 April 2024. Date of publication 2 May 2024; date of current version 25 July 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62173013 and Grant 62073005; in part by the Beijing Natural Science Foundation under Grant 4232049 and Grant L233005; and in part by the Fundamental Research Funds for the Central Universities under Grant YWF-23-03-QB-015. *(Corresponding author: Haitao Yuan.)*

Jing Bi and Haisen Ma are with the School of Software Engineering, Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China (e-mail: bijing@bjut.edu.cn; mahaisen@emails.bjut.edu.cn).

Haitao Yuan is with the School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China (e-mail: yuan@buaa.edu.cn).

Rajkumar Buyya is with the Cloud Computing and Distributed Systems Laboratory, School of Computing and Information Systems, University of Melbourne, Melbourne, VIC 3010, Australia (e-mail: rbuyya@unimelb.edu.au).

Jinhong Yang is with CSSC Systems Engineering Research Institute, Beijing 100036, China (e-mail: yangjinhong.66@163.com).

Jia Zhang is with the Department of Computer Science, Southern Methodist University, Dallas, TX 75206 USA (e-mail: jiazhang@smu.edu).

MengChu Zhou is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102 USA (e-mail: zhou@njit.edu).

Digital Object Identifier 10.1109/JIOT.2024.3395610

I. Introduction

As COMPANIES and organizations increasingly rely on cloud computing infrastructure, cloud data centers (CDCs) are growing popular due to their high availability and flexibility [1], [2]. These CDCs provide a variety of software and hardware, including computing, storage, and network resources in a pay-as-you-go way [3]. Individuals or organizations can rent computing resources as cloud services according to their needs. Cloud service providers [4] can avoid wasting resources and save the cost of managing infrastructure. Current famous Internet companies like Google, Microsoft, and Amazon have almost countless computing devices. To maximize the utilization of these computing resources, they have established their CDCs. Their computing tasks generate resource usage time series, including CPU, memory, disk, network, and I/O [5]. However, the high volatility and nonlinearity of the time series may result in over or under provisioning of resources [2], [6], [7]. For example, simultaneously, a large influx of tasks can easily cause resource shortages [8]. During periods with a few tasks, such as midnight, idle server clusters can result in resource wastage. According to [9], the mean CPU utilization of the whole servers in Alibaba CDCs varies between 5% and 85%, showing considerable fluctuations. Thus, designing an accurate prediction method that can effectively extract relationships and features among multidimensional resource usage time series is critically important.

Time series prediction has attracted a considerable number of studies [10]. Traditional prediction approaches include linear regression [11] and autoregressive integrated moving average (ARIMA) [12], [13], [14]. Nevertheless, when the regularity of the time series is not obvious, most of them cannot achieve the accurate prediction. In addition, these

approaches fail to extract complicated characteristics and patterns of time series data sets efficiently. Unlike the abovementioned methods, recurrent neural network (RNN) models have stronger sequence processing capabilities. Their variants [15], [16], [17], [18], [19] have been thoroughly employed for the time series prediction in the past few years. For example, long short-term memory (LSTM) is adopted to predict future short-term wind power [15]. Gupta et al. propose a sparse bidirectional LSTM (BiLSTM) network for future resource usage prediction. Saha et al. choose the LSTM-based encoder and decoder for the multistep Internet traffic prediction.

However, they cannot efficiently capture long-term dependencies and association information among different dimensions in the time series. Currently, some studies [20], [24], [25], [26], [27] have used transformer-based models to achieve the prediction. For example, a variant named multisize patched spatial-temporal transformer is presented to achieve the urban crowd prediction in [20]. A nonautoregressive transformer-based model is designed for vehicle trajectory prediction. A variant that combines the transformer and the Markov-chain Monte Carlo algorithm is designed to predict electrical energy consumption. However, these studies cannot effectively extract the frequency domain information in the series. To solve the abovementioned challenges, we design a Forecasting method based on the Integration of a Savitzky–Golay (SG) filter [21], a frequency enhanced decomposed transformer (FEDformer) model [22], and a frequency-enhanced channel attention mechanism (FECAM) [23], named FISFA for short. FISFA first adopts the SG filter to reduce noise and smooth the raw time series of resources. Then, FISFA adopts the FEDformer model to accurately predict resource usage time series by capturing their global features. In addition, FISFA adopts the FECAM module to improve the capability for extracting frequency features. Our key contributions are summarized as follows.

1) This work innovatively applies a noise reduction method of the filter of SG. It can smooth extreme points in the time series, highlight critical features of the data, and facilitate subsequent learning and extraction of features.
2) This work designs an improved transformer-based model integrating FEDformer and FECAM to achieve higher forecasting accuracy of resource usage series. The proposed method can learn frequency domain information and relationships among the multidimensional time series of resources.
3) This work designs a new hybrid metaheuristic algorithm, i.e., genetic simulated annealing-based particle swarm optimization (GSPSO) to optimize the setting of hyperparameters. GSPSO integrates quick convergence of PSO, global search ability of simulated annealing (SA), and diversity of genetic algorithm (GA).

The remainder of this article is structured as follows. Section II discusses the related work. Section III describes the framework of FISFA. Experimental results are presented in Section IV. Section V concludes this article along with a discussion on future work..

II. Related Work

Recent studies have proposed predicting computing resources. These prediction methods mainly include two kinds: 1) classical and 2) deep learning-based prediction methods.

A. Classical Time Series Prediction Methods

Traditional statistical analysis and regression methods are employed in resource usage forecasting in cloud computing. Gyeera et al. [28] adopted a boosted decision tree (BDT) regression method in a realistic testbed in the Azure cloud. BDT performs better than other machine learning algorithms, such as stochastic gradient descent and ordinary least square linear regression. However, as an iterative algorithm, BDT requires a long training time. Zhang et al. [29] proposed an XGBoost-based lane change prediction method with the realistic series data collected from autopilot vehicles. It has higher forecasting accuracy than adaptive boosting, gradient boosting trees, and random forest. However, the data set used in this work is less volatile than the sequence of resources in CDCs. An integrated prediction method that combines seasonal ARIMA (SARIMA) and gradient BDT is designed in [30]. However, SARIMA suffers from significant errors in long-term prediction. Besides, it performs poorly in capturing nonlinear characteristics of the sequence data. Shen and Yan [31] presented a support vector machine (SVM)-based transfer method for predicting rolling bearing remaining useful life. Yet, its data set shows lower fluctuation than the resource usage series in real-life large-scale CDCs. Wang and Li [32] introduced an enhanced linear regression algorithm for the prediction of real-time CPU temperature of servers. However, it fails to capture certain potential features in the data, mainly when dealing with highly nonlinear and nonstationary series. To achieve an online workload prediction framework, Kim et al. [33] proposed an ensemble model using several traditional prediction methods, including linear regression, linear SVM, ARIMA, etc. It achieves higher accuracy than a single traditional forecasting method.

Above all, most classic time series forecasting models are based on statistical or regression models. These methods require apparent trends in the time series and perform poorly in long-term prediction. Unlike these methods, this work employs an improved transformer-based model, which can extract features from cloud environments' highly nonlinear and variable computing resource usage data.

B. Deep Learning-Based Prediction Methods

With the improvement of the computing power of servers, many studies utilize deep learning models for addressing time series prediction problems. Kumar et al. [34] proposed a workload forecasting approach for requests for the industrial Internet of Things (IoT). This approach adopts deep autoencoders (DAEs) to predict the CPU cycles of cloud servers. However, DAEs train each layer individually in a layerwise manner. It suffers from long training time and high-computational complexity. Li et al. [35] introduced a temporal convolutional network (TCN)-based prediction

model for utility-scale photovoltaic forecasting. It captures spatial-temporal correlations to improve prediction accuracy for enormous intrahour photovoltaic power. Wang et al. [36] designed a prediction model that consists of a TCN layer and a graph convolution network (GCN) layer for traffic data sets of geo-distributed data centers. Nevertheless, it mainly focuses on the temporal dependencies of the extracted series. A deep concatenated multilayer perceptron [37] is proposed in an IoT network for fog sensor data prediction. However, the multilayer perceptron network often yields inferior prediction outputs than standard LSTM models. Ruan et al. [38] presented a feature-enhanced LSTM approach to extract the crucial sequence patterns in the cloud environment. However, LSTM [39] is an RNN-based model that fails to handle the gradient vanishing issue during the training process effectively. Its performance is unsatisfying for long-term prediction. The emergence of the transformer model has revolutionized the conventional use of RNN structures for processing sequence data. The model employs an encoder–decoder architecture and depends on an attention mechanism. Gao et al. [40] introduced a dual transformer model to predict both lane change intentions and trajectory projections of target vehicles. However, the trajectory data set exhibits lower volatility than the workload data from CDCs. Furthermore, many variants of transformer-based models have been introduced and utilized in time series forecasting. Zhang et al. [41] proposed an improved Informer by a data augmentation approach for forecasting the deterioration of aircraft engines. However, the transformers perform better than RNNs in capturing long-term dependency in the time domain. These methods cannot effectively investigate the patterns in the frequency domain. Yet, the frequency domain information is crucial in forecasting data points in time series.

In summary, current deep learning methods mainly adopt RNNs and transformers for forecasting the time series. RNN-based methods fail to solve the problems of gradient vanishing and long-term prediction. Most transformer-based methods primarily focus on the temporal information within the time series while disregarding the crucial frequency domain information. Unlike previous studies, our work proposes a new method named FISFA that integrates the SG filter, FEDformer, and FECAM module for multidimensional prediction of the resource usage time series in CDCs. During the data preprocessing stage, the SG filter eliminates noises and outliers in the raw data. Then, the FEDformer with FECAM effectively captures the frequency domain information in the time series data, leading to a more accurate prediction.

III. Model Framework

The section describes the details of FISFA. First, our problem definition is shown in Section III-A. Furthermore, we present the filter of SG in Section III-B. Then, we describe FISFA in detail in Section III-C. Finally, we present the details of the GSPSO used to optimize FISFA's hyperparameter setting. For clarity, Table I summarizes the main abbreviations in this work.

TABLE I
Abbreviation List

Abbreviation	Definition
CDCs	Cloud Data Centers
ARIMA	AutoRegressive Integrated Moving Average
SVM	Support Vector Machine
BDT	Boosted Decision Tree
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
DAE	Deep AutoEncoder
GCN	Graph Convolution Network
TCN	Temporal Convolutional Network
SG	Savitzky-Golay filter
FEB	Frequency-Enhanced Block
FEA	Frequency-Enhanced Attention
DFT	Discrete Fouier Transform
DCT	Discrete Cosine Transform
FECAM	Frequency-Enhanced Channel Attention Mechanism
PSO	Particle Swarm Optimization
GSPSO	Genetic Simulated annealing-based PSO

A. Problem Definition

This work chooses I to represent a multidimensional computing resource usage series in a CDC and $I=(I_1, I_2, \ldots, I_{t-1}, I_t)$. Previous t time slots are used to predict the values of computing resource usage at $t+1$. $\hat{y}_{t+1}$ denotes the final prediction value, which is obtained as

$$\hat{y}_{t+1} = \mathrm{f}(I_1, I_2, \ldots, I_{t-1}, I_t). \tag{1}$$

The proposed method aims to reduce errors between the ground truth values and the predicted ones.

B. Savitzky–Golay Filter

The SG filter [21] can decrease the noise of the time series through the least square polynomial smoothing method. Thus, we adopt it to extract the primary information in the preprocessing phase. It is processed by fitting successive subsets of each single-dimensional resource usage series with a low-degree polynomial.

I_t is the value (CPU or memory usage) in time slot t. $P_k=(p_{k-b}, \ldots, p_k, \ldots, q_{k+b})$, $k\in[b+1, t-b]$, which is a subsequence of I. Its width is $2b+1$. The SG filter adopts the following polynomial to fit it:

$$O(n)=\sum_{r=0}^{R} a_r n^r \quad n\in[-b, b] \tag{2}$$

where a_r denotes coefficient r of the polynomial and R denotes a polynomial order. The fitting is achieved by minimizing the mean-squared error ϵ for each subsequence centered at 0. ϵ is defined as

$$\epsilon=\sum_{n=-b}^{b}(O(n)-p_{k+b})^2=\sum_{n=-b}^{b}\left(\sum_{r=0}^{R} a_r n^r-p_{k+b}\right)^2. \tag{3}$$

The smoothed value is yielded by $O(n)$ at the central point $n=0$ and $O(0)=a_0$. The process above is iterated for each time slot.

C. FISFA Model

In addition to using the SG filter in the data preprocessing stage, FISFA combines the FEDformer model and the FECAM block. The details of the two modules are given below.

1) FEDformer: FEDformer [22] follows the encoder–decoder structure, which includes four modules: 1) frequency-enhanced block (FEB); 2) frequency-enhanced attention (FEA); 3) mixture of experts decomposition block (MOEDecomp); and 4) a feed-forward layer. The encoder is defined as χ_{en}^{l}=Encoder(χ_{en}^{l-1}). χ_{en}^{l} is the output in the encoder layer l, $l\in\{1,\ldots,N\}$. N denotes the number of layers in encoder. $\chi_{en}^{0}\in\mathbb{R}^{I\times D}$ denotes the embedded result of the historical time series. D denotes the dimension of the embedding layer. Encoder($\cdot$) is defined as

$$\begin{aligned} S_{en}^{l,1},\_ &= \text{MOEDecomp}\left(\text{FEB}(\chi_{en}^{l-1})+\chi_{en}^{l-1}\right)\\ S_{en}^{l,2},\_ &= \text{MOEDecomp}\left(\text{FeedForward}(S_{en}^{l,1})+S_{en}^{l,1}\right)\\ \chi_{en}^{l} &= S_{en}^{l,2} \end{aligned} \tag{4}$$

where $S_{en}^{l,i}$, $i\in\{1,2\}$ is the seasonal component after decomposition block i in layer l. The symbol _ means the eliminated trend part. FEB is implemented based on discrete fourier transform (DFT). It can effectively replace the self-attention block in traditional transformer models. Given a series of numbers, X_n, in the time domain, FEB processes it with the Fourier transform and the inverse Fourier transform. In this way, the conversion between the time domain and the frequency domain is realized. X_ς is a complex series in the frequency domain, and $1\le\varsigma\le\hat{\varsigma}$ where $\hat{\varsigma}$ denotes the sequence length of complex numbers in the frequency domain. X_n is the value of time point n (n=0, 1, …, $t-1$) in the time series of real numbers in the time domain. X_ς and X_n are defined as

$$X_\varsigma = \sum_{n=0}^{t-1} X_n e^{-i\varsigma\omega n} \tag{5}$$

$$X_n = \sum_{\varsigma=0}^{\hat{\varsigma}-1} X_\varsigma e^{i\varsigma\omega n} \tag{6}$$

where i denotes the imaginary unit and ω denotes the angular frequency.

The output of decoder layer l includes Γ_{de}^{l} and χ_{de}^{l}, which are the results of Decoder(χ_{en}^{l-1}, Γ_{de}^{l-1}). The Decoder($\cdot$) is defined as

$$\begin{aligned} S_{de}^{l,1},\Gamma_{de}^{l,1} &= \text{MOEDecomp}\left(\text{FEB}(\chi_{de}^{l-1})+\chi_{de}^{l-1}\right)\\ S_{de}^{l,2},\Gamma_{de}^{l,2} &= \text{MOEDecomp}\left(\text{FEA}(S_{de}^{l,1},\chi_{de}^{N})+S_{de}^{l,1}\right)\\ S_{de}^{l,3},\Gamma_{de}^{l,3} &= \text{MOEDecomp}\left(\text{FeedForward}(S_{de}^{l,2})+S_{de}^{l,2}\right)\\ \chi_{de}^{l} &= S_{de}^{l,3}\\ \Gamma_{de}^{l} &= \Gamma_{de}^{l-1}+W_{l,1}\Gamma_{de}^{l,1}+W_{l,2}\Gamma_{de}^{l,2}+W_{l,3}\Gamma_{de}^{l,3} \end{aligned} \tag{7}$$

where $S_{de}^{l,i}$ and $\Gamma_{de}^{l,i}$, $i\in\{1,2,3\}$, denote the seasonal and trend components after the decomposition block i in layer l, respectively. $W_{l,i}$ is the projector. FEA is also implemented based on DFT with an attention mechanism. It can replace the cross-attention block. MOEDecomp is a progressive decomposition architecture. Traditional fixed-window averaging pooling struggles to extract trends effectively. Thus, it comprises a set of average filters with varying sizes designed to extract multiple trend components in the input signal. Additionally, it utilizes many data-dependent weights to merge these components as the ultimate trend, $\acute{X}_{\text{trend}}$, which is a time series decomposed by the MOEDecomp operation and $\acute{X}_{\text{trend}}$ includes $\Gamma_{de}^{l,1}$, $\Gamma_{de}^{l,2}$, and $\Gamma_{de}^{l,3}$. $\acute{X}_{\text{trend}}$ is defined as

$$\acute{X}_{\text{trend}}=\text{Softmax}(\text{Linear}(\psi)G(\psi)) \tag{8}$$

where ψ denotes the input of the MOEDecomp operation in (7), Linear(ψ) denotes the linear operation on ψ, $G(\psi)$ denotes the average pooling filtering operation on ψ, and Softmax(Linear(ψ)) is the weighted result for mixing extracted trends, which is the final $\acute{X}_{\text{trend}}$.

Finally, the prediction results are obtained by the sum of two decomposed components, i.e., $W\chi_{de}^{M}+\Gamma_{de}^{M}$. M denotes the number of layers in the decoder. W is used to convert the seasonal component χ_{de}^{M} to the target dimension.

2) FECAM Block: Current methods mainly adopt the Fourier transform to extract frequency information from the time series. If the values of the two ends of the sequence differ greatly, the Fourier transform introduces high-frequency noise. It causes an error for boundary information called the Gibbs phenomenon. To address this problem, FECAM [23] based on discrete cosine transform (DCT) is proposed. FECAM adopts DCT to extract the frequency information. This method avoids the Gibbs issue and the operation of inverse transformation. The features are divided by FECAM into d subgroups, i.e., $[\kappa_1,\kappa_2,\ldots,\kappa_d]$ according to the dimension of the input. Each subgroup is processed by the component of DCT from low frequency to high one. F^z denotes the zth frequency channel vector, which is obtained as

$$F^z=\text{DCT}_j(\kappa^z),\ z\in\{0,1,\ldots,d\} \tag{9}$$

where DCT_j denotes the frequency component corresponding to κ^z. The stack operation obtains the complete frequency channel vector F

$$F=\text{stack}\left(F^0,F^1,\ldots,F^{d-1}\right). \tag{10}$$

Finally, critical temporal information from the frequency domain of each channel feature is obtained. Therefore, as illustrated in Fig. 1, the filter of SG is adopted to denoise the raw data. Then, we use the FEDformer model to analyze the context information in the time series. Besides, we add a FEACM module between the encoder and the decoder. It additionally boosts the capacity to extract the frequency information in the time series.

D. GSPSO

Deep learning-based models typically have many hyperparameters that highly affect the performance of the FISFA. For example, they involve the number of train epochs, batch size, the layer numbers in the encoder and decoder, learning rate, and dropout rate. Tuning these hyperparameters is a time-consuming task. PSO [42], [43] can be used to determine

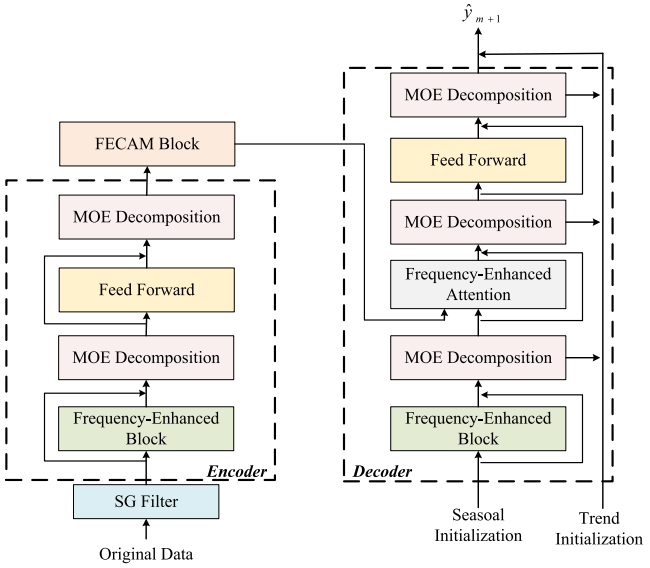

Fig. 1. Structure of FISFA.

Algorithm 1 GSPSO

1: Initialize particle information randomly
2: Update $\check{\phi}$ of particles
3: Select $\check{x}_i$ and $\hat{x}$
4: Set GA's θ_5, SA's θ_4^1 and θ_7, and PSO's parameters, including $\check{\theta}_2$, $\hat{\theta}_2$, θ_3, $\hat{\theta}_1$, $\check{\theta}_1$, $\hat{\theta}_6$, $\hat{g}$, and $|x|$
5: $g \leftarrow 1$
6: **while** $\theta_6 \leq \hat{\theta}_6$ and $g \leq \hat{g}$ **do**
7: Execute crossover of GA on $\check{x}_i$ and $\hat{x}$ to yield an offspring $\check{x}_i$
8: Execute mutation of GA on each bit of $\check{x}_i$ with a probability θ_5
9: Execute selection of GA for particle i
10: Calculate velocities of particles with (11)
11: Calculate positions of particles with (12) and (13)
12: Calculate $\check{\phi}$ of particles
13: Change $\check{x}_i$ of particle i, and $\hat{x}$
14: $\theta_4^g \leftarrow \theta_4^g \cdot \theta_7$
15: $\theta_1 \leftarrow \left(\hat{\theta}_1 - \check{\theta}_1\right) \cdot \frac{\hat{g}-g}{\hat{g}} + \check{\theta}_1$
16: Update θ_6 of particles with the same $\check{\phi}$
17: $g \leftarrow g+1$
18: **end while**
19: **return** $\hat{x}$

the optimal hyperparameters effectively. Our work designs an improved version of PSO [44] to accomplish the optimal hyperparameters.

Similar to social behaviors of bird or fish swarm [45], [46], PSO involves a population of particles moving through a search space. These particles adjust positions based on their individual experiences and those of neighboring particles to find the optimal solution, and therefore, they can converge quickly. However, it often converges toward local optima when applied to address constrained problems with sophisticated solution spaces. Besides, SA employs the rule of Metropolis acceptance, thus enabling moves that might deteriorate the search. This capability allows SA to converge toward the global optima using the optimal cooling rate. Nevertheless, it is worth noting that SA converges slowly. Besides, in GA, genetic operations yield diverse individuals, enhancing the global search capability. Therefore, GSPSO combines the strengths of three algorithms by integrating the rule of Metropolis acceptance, genetic operations, and PSO.

Algorithm 1 exhibits GSPSO's pseudocodes. Line 1 randomly sets the position and velocity of each particle. Line 2 calculates each particle's fitness value $\check{\phi}$. Line 3 updates $\check{x}_i$ and $\hat{x}$. $\check{x}_i$ means particle i's locally optimal position. $\hat{x}$ means the globally optimal position in the population. Line 4 sets GA's mutation possibility θ_5, SA's initial temperature θ_4^1 and its cooling rate θ_7, and PSO's parameters, including $\check{\theta}_2$, $\hat{\theta}_2$, θ_3, $\hat{\theta}_1$, $\check{\theta}_1$, $\hat{\theta}_6$, $\hat{g}$, and $|x|$. $\check{\theta}_2$ means an individual coefficient. $\hat{\theta}_2$ means a social acceleration coefficient. θ_3 means the coefficient of acceleration for a superior particle. $\check{\theta}_1$ denotes the inertia weight. $\hat{g}$ denotes the total iteration number. $\hat{\theta}_6$ means the percentage of particles with identical $\check{\phi}$. $|x|$ is the size of population. Line 6 means the while loop stops if $g > \hat{g}$ or $\theta_6 > \hat{\theta}_6$. Line 7 executes GA's crossover on $\check{x}_i$ and $\hat{x}$ with the single-point crossover to yield an offspring $\check{x}_i$. Line 8 executes GA's mutation on each bit of offspring $\check{x}_i$ with a probability θ_5. Line 9 executes GA's selection to specify $\check{x}_i$ or $\dot{x}_i$ is chosen. $\dot{x}_i$ denotes the position of a superior particle for particle i. Line 10 updates the velocity of each particle with

$$v_i = \theta_1 v_i + \theta_3 w_3 \left(\dot{x}_i - x_i^g\right) \tag{11}$$

where v_i is the velocity of each particle i. x_i^g means particle i's position in iteration g. Line 11 changes the position of each particle with (12) and (13). More specifically, if $\tilde{\phi}(x_i^{g+1}) \leq \tilde{\phi}(x_i^g)$, x_i^{g+1} is selected; otherwise, it is conditionally selected if (13) is met

$$x_i^{g+1} = x_i^g + v_i \tag{12}$$

$$e^{\left(\frac{\tilde{\phi}(x_i^{g+1}) - \tilde{\phi}(x_i^g)}{\theta_4^g}\right)} > w_4 \tag{13}$$

where w_4 is a constant randomly selected in $(0, 1)$. θ_4^g is current temperature in iteration g.

Line 12 calculates each particle's fitness value $\check{\phi}$. Line 13 changes particle i's locally optimal position $\check{x}_i$ and the population's globally optimal position $\hat{x}$. Besides, θ_4^1 is the initial temperature, and θ_7 is its cooling rate. Line 14 reduces temperature by θ_7. $\hat{\theta}_1$ and $\check{\theta}_1$ are upper and lower bounds of inertia weight θ_1. Line 15 linearly decreases θ_1 from $\hat{\theta}_1$ to $\check{\theta}_1$. Line 16 calculates percentage θ_6 of particles with identical $\check{\phi}$. Line 19 returns $\hat{x}$, including the final setting of hyperparameters. Fig. 2 shows the flowchart of GSPSO to optimize the setting of several hyperparameters of FISFA, yielding the optimal hyperparameter setting that minimizes the training loss.

Moreover, GSPSO revises the optimal local position for each particle and updates the globally optimal position within the whole population. Additionally, the inertia weight and current temperature decrease linearly. Eventually, it adjusts

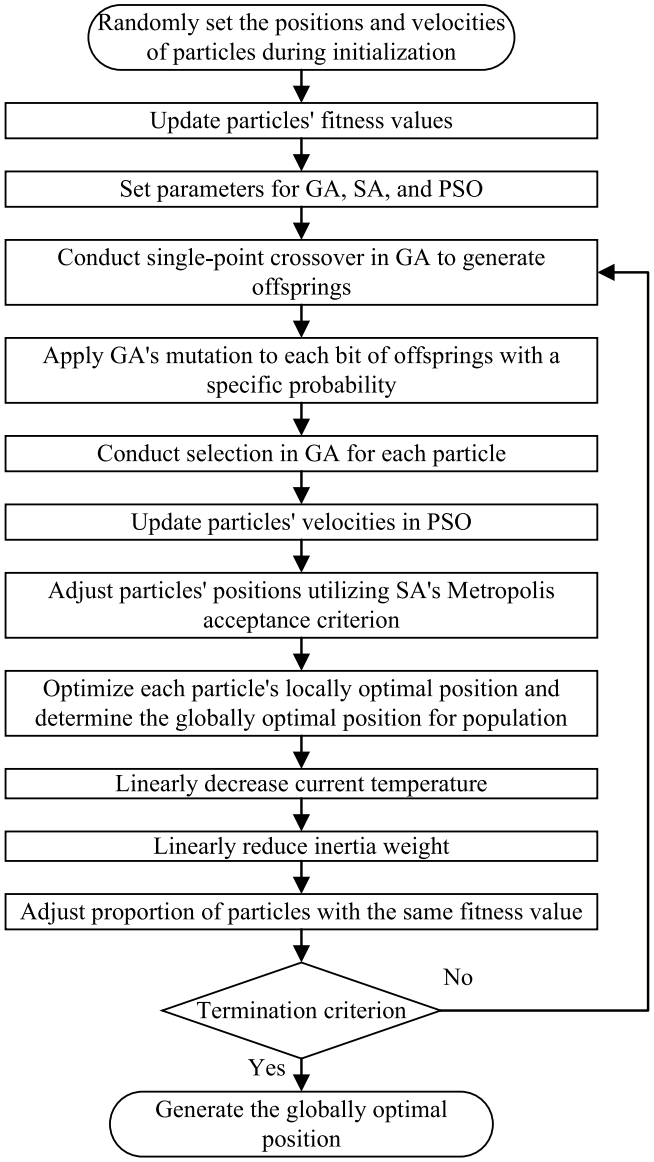

Fig. 2. Flowchart of GSPSO.

the proportion of particles sharing identical fitness values and determines whether the termination criterion is satisfied. If it is met, the globally optimal solution is attained; otherwise, the single-point crossover of GA and the following procedures continue to iterate until the termination criterion is satisfied.

E. Complexity Analysis

The most time-consuming operation of FISFA lies in the training stage. t is also the length of the input series. The time complexity $O(t^2)$ of transformer mainly comes from the self-attention mechanism. FISFA replaces the self-attention mechanism with the DFT with a time complexity $O(t(\log t))$ [22] for the frequency domain feature extraction. Meanwhile, GSPSO performs $\hat{g}$ iterations. Therefore, the time complexity of FISFA is $O(\hat{g}t(\log t))$.

IV. Performance Evaluation

We assess FISFA with realistic data sets and compare its performance with transformer-based prediction models and other traditional methods.

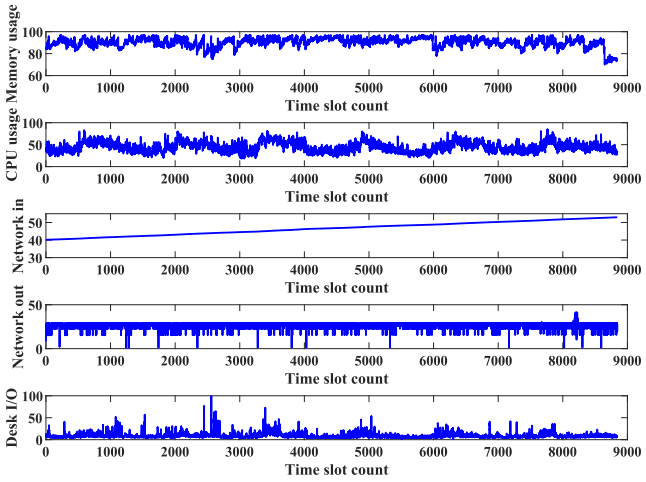

Fig. 3. Resource usage time series of machine ID 649 in the Alibaba cluster data set.

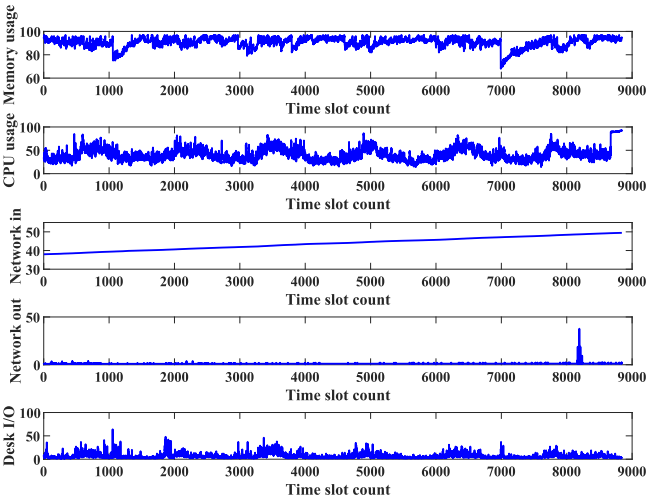

Fig. 4. Resource usage time series of machine ID 1932 in the Alibaba cluster data set.

A. Data Set and Experimental Setup

To confirm the efficacy of FISFA, we adopt two heterogeneous real-world data sets collected from Alibaba and Google clusters, respectively. The former data set includes runtime information on machine resource usage from 4,000 machines in eight days. The log of Cluster-trace-v2018 of Alibaba provides seven cluster data tables. The machine usage table includes CPU utilization, memory utilization, memory bandwidth, cache miss per thousand instructions, incoming and outgoing network traffic, and disk I/O. We select five key resource metrics for the prediction. The time interval is 1 min. Tasks are categorized based on the machines with IDs 649 and 1932. Finally, the resource usage time series is obtained and shown in Figs. 3 and 4. Google cluster traces provide information about CDCs in eight regions in May 2019. We choose one data set with a timezone located in New York, USA. It includes information about CPU usage and alloc sets (shared resource reservations used by jobs). We split 31 days into 14 880 3-min time slots. Finally, the time series of CPU and memory resources requested for the instance are shown in Fig. 5.

TABLE II
COMPARISON OF DIFFERENT COMBINATIONS OF HYPERPARAMETERS

Combinations	Layer # in encoder (α)	Layer # in decoder (β)	Batch size (γ)	RSME	MAE	MAPE
Combination 1	1	1	32	2.35794	1.38672	0.07374
Combination 2	2	1	32	2.34254	1.37809	0.07331
Combination 3	2	1	16	2.35916	1.39187	0.07390
Combination by GSPSO	1	2	16	**2.33819**	**1.37100**	**0.07325**

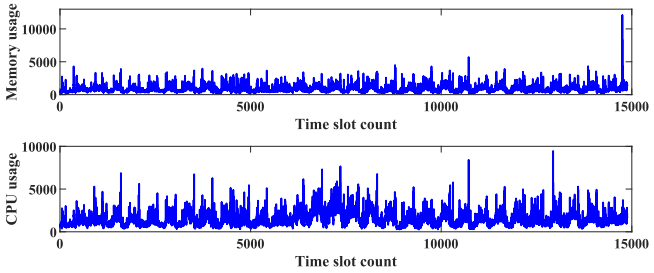

Fig. 5. Resource usage time series of the Google cluster data set.

TABLE III
SETTING OF FISFA PARAMETERS

Parameters	Values
Dimension of model	256
Learning rate	0.001
Activation	gelu
Layer number in encoder	1
Layer number in decoder	2
Batch size	16
Early stopping patience	9
Loss function	MSE

TABLE IV
COMPARISON OF FEDFORMER AND FEDFORMER WITH FECAM

Dimension of model	RMSE	MAE	MAPE
8	3.11518	1.79656	0.09408
	3.11**257**	1.79**477**	0.09**400**
16	3.07927	1.78386	0.08947
	3.07**812**	1.78**276**	0.08943
32	3.03130	1.72264	0.09138
	3.031**13**	1.722**50**	0.09138
64	2.99807	1.69626	0.09013
	2.99**768**	1.69**571**	0.090**08**
128	2.99799	1.69064	0.08947
	2.99805	1.690**59**	0.08947
256	2.99309	1.68770	0.08900
	2.9**8965**	1.68**498**	0.08**886**

B. Evaluation Metrics

We utilize three metrics, including i.e., root mean square error (RMSE) [47], mean absolute percentage error (MAPE) [48], and mean absolute error (MAE) [49]. They are calculated as follows:

$$\begin{aligned} \text{RMSE} &= \sqrt{\frac{1}{m}\sum_{t=1}^{m}\left(y_t-\hat{y}_t\right)^2} \\ \text{MAPE} &= \frac{100\%}{m}\sum_{t=1}^{m}\left|\frac{y_t-\hat{y}_t}{y_t}\right| \\ \text{MAE} &= \frac{1}{m}\sum_{t=1}^{m}\left|y_t-\hat{y}_t\right| \end{aligned} \tag{14}$$

where n is the sample number, $\hat{y}_t$ is the average of the ground truth values, and y_t is the predicted result in time slot t.

C. Hyperparameter Setting

To determine the optimal hyperparameter setting, comprehensive experiments are conducted systemically. Table III displays the final hyperparameter tuning results. The rate of learning is 0.001. The model dimension is 256. The function of loss is MSE, and the early stopping patience is 9. Besides, we utilize GSPSO to optimize several key hyperparameters in our model. Three crucial hyperparameters are chosen, including the layer number of encoder (α), the layer number of decoder (β), and the batch size (γ). Finally, α, β, and γ are set to 1, 2, and 16, respectively. Table II illustrates experimental results after selecting four different combinations of hyperparameters. The results show that the hyperparameter configuration yielded by GSPSO produces the highest prediction accuracy. The parameter configurations for FISFA are outlined in Table III.

D. Analysis of Prediction Results

We allocate 70% of the time series for the training, 10% for the validation, and the remaining 20% for the testing. To evaluate FECAM in the prediction, comparison experiments of FEDformers with and without the FECAM block are conducted. Table IV shows the results of three evaluation metrics. The odd and even rows represent metric values for FEDformer and FEDformer with FECAM, respectively. The results prove that the FEDformer with FECAM outperforms its vanilla version.

We choose several benchmark methods to compare our FISFA with its other state-of-the-art peers comprehensively. For example, LSTM is based on the gated cell and is commonly used for time series prediction. However, it suffers from the gradient explosion problem during training and cannot effectively extract the correlation among multidimensional data. Informer is an improved transformer model with low-time complexity and memory utilization. However, it cannot effectively extract frequency domain features.

Furthermore, Tables V–VII show the performance comparison between FISFA and various prediction methods, including LSTM and transformer-based models, e.g., transformer and Informer. The abbreviation FEC signifies that forecasting models employ FECAM. SG- means that the SG filter is adopted. Table V shows the transformer-based models achieve higher performance than LSTM in the multidimensional prediction. FECAM and the SG filter improve the evaluation metric

TABLE V
PERFORMANCE COMPARISON OF ALL METHODS WITH THE DATA SET OF MACHINE ID 649 FROM ALIBABA

Methods	RMSE	MAE	MAPE
LSTM	3.46993	2.26280	0.10308
Transformer	3.09399	1.89245	0.10084
Informer	3.19903	2.02076	0.10897
FEDformer	2.99309	1.68770	0.08900
Transformer+FEC	3.08164	1.89849	0.10042
Informer+FEC	3.19222	2.00831	0.10858
FEDformer+FEC	2.98965	1.68498	0.08886
SG-LSTM	3.43238	2.22098	0.10182
SG-Transformer	*2.40196*	*1.45422*	*0.07335*
SG-Informer	2.47797	1.54587	0.07423
FISFA	**2.32237**	**1.34573**	**0.07271**

TABLE VI
PERFORMANCE COMPARISON OF ALL METHODS WITH THE DATA SET OF MACHINE ID 1932 FROM ALIBABA

Methods	RMSE	MAE	MAPE
LSTM	3.97746	2.23827	0.27784
Transformer	3.37595	1.77665	0.10875
Informer	3.53577	1.85022	0.11172
FEDformer	3.33543	1.39523	0.11575
Transformer+FEC	3.41701	1.79651	0.11500
Informer+FEC	3.33480	1.78247	0.10704
FEDformer+FEC	3.31267	1.39752	0.11484
SG-LSTM	3.95284	2.21583	0.27527
SG-Transformer	*2.46992*	*1.28261*	*0.09220*
SG-Informer	2.51991	1.33650	0.08498
FISFA	**2.29162**	**1.01012**	**0.08320**

TABLE VII
PERFORMANCE COMPARISON OF ALL METHODS WITH THE GOOGLE DATA SET

Methods	RMSE	MAE	MAPE
LSTM	491.185	297.653	0.27746
Transformer	481.125	299.474	0.27375
Informer	488.582	306.436	0.27840
FEDformer	476.954	296.782	0.26898
Transformer+FEC	482.027	300.138	0.27428
Informer+FEC	484.144	301.987	0.27642
FEDformer+FEC	476.818	295.825	0.26754
SG-LSTM	396.026	242.718	0.23827
SG-Transformer	400.740	*249.598*	*0.22345*
SG-Informer	*398.055*	250.062	0.22784
FISFA	**387.199**	**239.265**	**0.21533**

TABLE VIII
ABLATION STUDIES OF FISFA WITH THREE METHODS

Methods	RMSE	MAE	MAPE
w/o SG filter	2.98965	1.68498	0.08886
w/o FECAM	2.32727	1.34451	0.07337
w/o GSPSO	2.35916	1.39187	0.07390
FISFA	**2.32237**	**1.34573**	**0.07271**

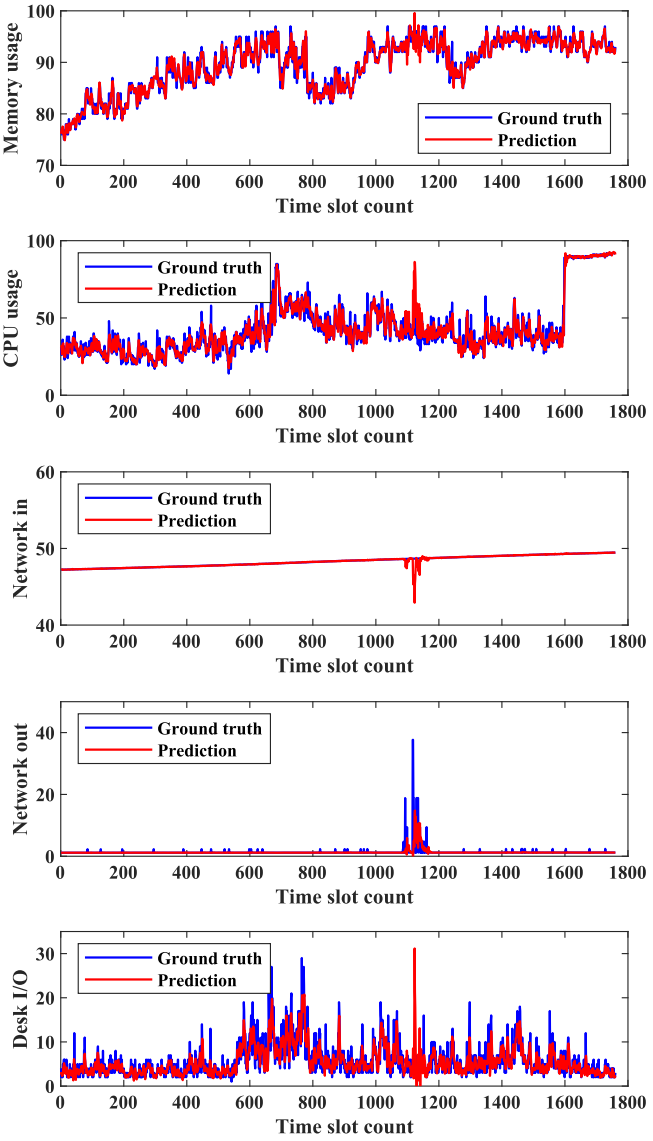

Fig. 6. Prediction curves of resource usage time series of machine ID 1932.

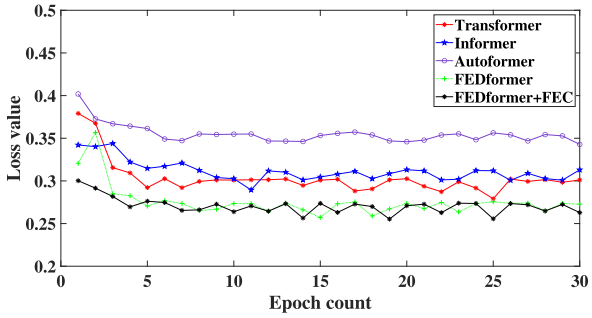

Fig. 7. Loss values of different methods for resource usage time series of machine ID 1932.

values, and FISFA achieves the highest accuracy among all these methods.

Table VIII shows the ablation studies of FISFA with three methods. It is evident that the addition of each method can bring improvement to the prediction. Fig. 6 shows ground truth values and the predicted ones of RAM usage, CPU usage, Network in, Network out, and Desk I/O, respectively. Fig. 7 compares loss values of transformer, Informer, Autoformer, FEDformer, and FEDformer with FECAM for the resource usage time series of machine ID 1932, respectively. Fig. 8 shows the loss values of different methods after adding the SG filter. After iteration 10, it is evident that FISFA's loss values are comparatively smaller than those of other models. This demonstrates that FISFA possesses superior modeling capabilities compared with other transformer variants. Consequently,

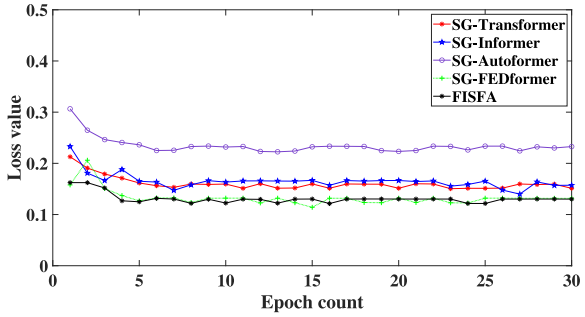

Fig. 8. Loss values of different methods after adding the SG filter.

FISFA outperforms other benchmark methods given the same setting.

V. Conclusion and Future Work

Current cloud providers face a critical but challenging problem of accurately predicting computing resource usage in CDCs. Resource usage series is often multidimensional and volatile. Each series is characterized by different trends, increasing the difficulty of forecasting. Most current forecasting methods cannot effectively extract correlations among multiple series and frequency domain information. This work proposes a Forecasting method based on the Integration of a SG filter, a FEDformer model, and a FECAM, named FISFA for short, for forecasting the multidimensional computing resource usage series. FISFA initially adopts the SG filter to accomplish better noise reduction. It designs a FEDformer model with a FECAM to investigate key patterns from resource usage time series in the frequency domain. In addition, a hybrid meta-heuristic optimization algorithm called genetic SA-based particle swarm optimizer is proposed to optimize key hyperparameters of FISFA. At last, experiments with two heterogeneous real-world data sets from Alibaba and Google demonstrate that FISFA achieves superior forecasting accuracy than its baseline peers. Against LSTM, transformer, and Informer, our prediction accuracy is improved by 32.14%, 25.49%, and 27.71%, respectively.

As part of future work, we will apply FISFA to more diverse real-world workload data sets. We also plan to incorporate novel spatial-temporal GCNs to enhance performance. In addition, we plan to employ meta-learning to provide beneficial guidance on learning a more generalized and adaptive model for predicting resource usage in CDCs.

References

[1] A. Jayanetti, S. Halgamuge, and R. Buyya, "Multi-agent deep reinforcement learning framework for renewable energy-aware workflow scheduling on distributed cloud data centers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 35, no. 4, pp. 604–615, Apr. 2024.

[2] R. Findeisen, F. Allgöwer, and L. T. Biegler, *Assessment and Future Directions of Nonlinear Model Predictive Control* (Lecture Notes in Control and Information Sciences), vol. 358, Berlin, Germany: Springer, 2007.

[3] Y. Xia, M. Zhou, X. Luo, Q. Zhu, J. Li, and Y. Huang, "Stochastic modeling and quality evaluation of infrastructure-as-a-service clouds," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 162–170, Jan. 2015.

[4] H. Yuan, J. Bi, and M. Zhou, "Profit-sensitive spatial scheduling of multi-application tasks in distributed green clouds," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 3, pp. 1097–1106, Jul. 2020.

[5] Q. Peng, C. Wu, Y. Xia, Y. Ma, X. Wang, and N. Jiang, "DoSRA: A decentralized approach to online edge task scheduling and resource allocation," *IEEE Internet Things J.*, vol. 9, no. 6, pp. 4677–4692, Mar. 2022.

[6] J. Bi, H. Yuan, K. Zhang, and M. Zhou, "Energy-minimized partial computation offloading for delay-sensitive applications in heterogeneous edge networks," *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 4, pp. 1941–1954, Oct.–Dec. 2022.

[7] H. Yuan, J. Bi, and M. Zhou, "Geography-aware task scheduling for profit maximization in distributed green data centers," *IEEE Trans. Cloud Comput.*, vol. 10, no. 3, pp. 1864–1874, Jul.–Sep. 2022.

[8] T. Long et al., "A deep deterministic policy gradient-based method for enforcing service fault-tolerance in MEC," *Chin. J. Electron.*, vol. 33, no. 5, pp. 1–11, 2023.

[9] J. Guo et al., "Who limits the resource efficiency of my datacenter: An analysis of Alibaba datacenter traces," in *Proc. IEEE/ACM 27th IEEE/ACM Int. Symp. Qual. Service*, 2019, pp. 1–10.

[10] S. Tuli, S. S. Gill, P. Garraghan, R. Buyya, G. Casale, and N. R. Jennings, "START: Straggler prediction and mitigation for cloud computing environments using encoder LSTM networks," *IEEE Trans. Services Comput.*, vol. 16, no. 1, pp. 615–627, Jan./Feb. 2023.

[11] X. Tang, X. Liao, J. Zheng, and X. Yang, "Energy efficient job scheduling with workload prediction on cloud data center," *Clust. Comput.*, vol. 21, pp. 1581–1593. Feb. 2018.

[12] B. Fei, X. Zhu, D. Liu, J. Chen, W. Bao, and L. Liu, "Elastic resource provisioning using data clustering in cloud service platform," *IEEE Trans. Services Comput.*, vol. 15, no. 3, pp. 1578–1591, May 2022.

[13] N. Hogade and S. Pasricha, "A survey on machine learning for geo-distributed cloud data center management," *IEEE Trans. Sustain. Comput.*, vol. 8, no. 1, pp. 15–31, Jan. 2023.

[14] Z. Xing, Y. He, X. Wang, K. Shao, and J. Duan, "VMD-IARIMA-based time-series forecasting model and its application in dissolved gas analysis," *IEEE Trans. Dielectr. Electr. Insul.*, vol. 30, no. 2, pp. 802–811, Apr. 2023.

[15] M. Hossain et al., "Optimized forecasting model to improve the accuracy of very short-term wind power prediction," *IEEE Trans. Ind. Informat.*, vol. 19, no. 10, pp. 10145–10159, Oct. 2023.

[16] S. Gupta, A. D. Dileep, and T. A. Gonsalves, "Online sparse BLSTM models for resource usage prediction in cloud datacentres," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 4, pp. 2335–2349, Dec. 2020.

[17] A. Zollanvari, K. Kunanbayev, S. Bitaghsir, and M. Bagheri, "Transformer fault prognosis using deep recurrent neural network over vibration signals," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–11, 2021.

[18] Y. Yu, G. Hu, C. Liu, J. Xiong, and Z. Wu, "Prediction of solar irradiance one hour ahead based on quantum long short-term memory network," *IEEE Trans. Quantum Eng.*, vol. 4, pp. 1–15, Apr. 2023.

[19] S. Saha, A. Haque, and G. Sidebottom, "Analyzing the impact of outlier data points on multi-step Internet traffic prediction using deep sequence models," *IEEE Trans. Netw. Service Manag.*, vol. 20, no. 2, pp. 1345–1362, Jun. 2023.

[20] Y. Xie, J. Niu, Y. Zhang, and F. Ren, "Multisize patched spatial-temporal transformer network for short- and long-term crowd flow prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 11, pp. 21548–21568, Nov. 2022.

[21] A. Savitzky and M. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Anal. Chem.*, vol. 36, pp. 1627–1639, Jul. 1964.

[22] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *Proc. 39th Int. Conf. Mach. Learn.*, 2022, pp. 1–19.

[23] M. Jiang, P. Zeng, K. Wang, H. Liu, W. Chen, and H. Liu, "FECAM: Frequency enhanced channel attention mechanism for time series forecasting," *Adv. Eng. Informat.*, vol. 58, pp. 1–11, Oct. 2023.

[24] C. Yang and Z. Pei, "Long-short term spatio-temporal aggregation for trajectory prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 4, pp. 4114–4126, Apr. 2023.

[25] X. Chen, H. Zhang, F. Zhao, Y. Cai, H. Wang, and Q. Ye, "Vehicle trajectory prediction based on intention-aware non-autoregressive transformer with multi-attention learning for Internet of Vehicles," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–12, Jul. 2022.

[26] H. Shen et al., "Electric vehicle velocity and energy consumption predictions using transformer and Markov-chain Monte Carlo," *IEEE Trans. Transp. Electrif.*, vol. 8, no. 3, pp. 3836–3847, Sep. 2022.

[27] P. Xie et al., "Spatio-temporal dynamic graph relation learning for urban metro flow prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 10, pp. 9973–9984, Oct. 2023.
[28] T. W. Gyeera, A. J. H. Simons, and M. Stannett, "Regression analysis of predictions and forecasts of cloud data center KPIs using the boosted decision tree algorithm," *IEEE Trans. Big Data*, vol. 9, no. 4, pp. 1071–1085, Aug. 2023.
[29] Y. Zhang, X. Shi, S. Zhang, and A. Abraham, "An XGBoost-based lane change prediction on time series data using feature engineering for autopilot vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 19187–19200, Oct. 2022.
[30] B. Feng, Z. Ding, and C. Jiang, "FAST: A forecasting model with adaptive sliding window and time locality integration for dynamic cloud workloads," *IEEE Trans. Services Comput.*, vol. 16, no. 2, pp. 1184–1197, Mar./Apr. 2023.
[31] F. Shen and R. Yan, "A new intermediate-domain SVM-based transfer model for rolling bearing RUL prediction," *IEEE/ASME Trans. Mechatronics*, vol. 27, no. 3, pp. 1357–1369, Jun. 2022.
[32] N. Wang and J.-Y. Li, "Efficient multi-channel thermal monitoring and temperature prediction based on improved linear regression," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–9, 2022.
[33] I. K. Kim, W. Wang, Y. Qi, and M. Humphrey, "Forecasting cloud application workloads with cloudinsight for predictive resource management," *IEEE Trans. Cloud Comput.*, vol. 10, no. 3, pp. 1848–1863, Jul.–Sep. 2022.
[34] M. Kumar, A. Kishor, J. K. Samariya, and A. Y. Zomaya, "An autonomic workload prediction and resource allocation framework for fog-enabled industrial IoT," *IEEE Internet Things J.*, vol. 10, no. 11, pp. 9513–9522, Jun. 2023.
[35] Y. Li, L. Song, S. Zhang, A. Komandur, and N. Lu, "A TCN-based hybrid forecasting framework for hours-ahead utility-scale PV forecasting," *IEEE Trans. Smart Grid*, vol. 14, no. 5, pp. 4073–4085, Sep. 2023.
[36] Z. Wang et al., "Large-scale measurements and prediction of DC-WAN traffic," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 5, pp. 1390–1405, May 2023.
[37] M. A. P. Putra, A. P. Hermawan, C. I. Nwakanma, D. S. Kim, and J. M. Lee, "FDPR: A novel fog data prediction and recovery using efficient DL in IoT networks," *IEEE Internet Things J.*, vol. 10, no. 19, pp. 16895–16906, Oct. 2023.
[38] L. Ruan et al., "Cloud workload turning points prediction via cloud feature-enhanced deep learning," *IEEE Trans. Cloud Comput.*, vol. 11, no. 2, pp. 1719–1732, Apr.–Jun. 2023.
[39] R. Zhang, J. Chen, Y. Song, Y. Wen, and C. Peng, "An effective transformation-encoding-attention framework for multivariate time series anomaly detection in IoT environment," *Mobile Netw. Appl.*, pp. 1–3, Aug. 2023.
[40] K. Gao et al., "Dual transformer based prediction for lane change intentions and trajectories in mixed traffic environment," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 6, pp. 6203–6216, Jun. 2023.
[41] Z. Zhang et al., "A data augmentation boosted dual Informer framework for the performance degradation prediction of aero-engines," *IEEE Sensors J.*, vol. 23, no. 11, pp. 12018–12030, Jun. 2023.
[42] Q. Ge et al., "Industrial power load forecasting method based on reinforcement learning and PSO-LSSVM," *IEEE Trans. Cybern.*, vol. 52, no. 2, pp. 1112–1124, Feb. 2022.
[43] H. Yan, Y. Qi, Q. Ye, and D. Yu, "Robust least squares twin support vector regression with adaptive FOA and PSO for short-term traffic flow prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14542–14556, Sep. 2022.
[44] J. Bi, H. Yuan, S. Duanmu, M. Zhou, and A. Abusorrah, "Energy-optimized partial computation offloading in mobile-edge computing with genetic simulated-annealing-based particle swarm optimization," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3774–3785, Mar. 2021.
[45] J. Bi, Z. Wang, H. Yuan, J. Qiao, J. Zhang, and M. Zhou, "Self-adaptive teaching-learning-based optimizer with improved RBF and sparse autoencoder for complex optimization problems," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 7966–7972.
[46] J. Bi et al., "Multi-swarm genetic Gray Wolf optimizer with embedded autoencoders for high-dimensional expensive problems," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 7265–7271.
[47] T. Chai and R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)?—Arguments against avoiding RMSE in the literature," *Geosci. Model Develop.*, vol. 7, pp. 1247–1250, Jun. 2014.
[48] J. Bi, L. Zhang, H. Yuan, and J. Zhang, "Multi-indicator water quality prediction with attention-assisted bidirectional LSTM and encoder–decoder," *Inf. Sci.*, vol. 625, pp. 65–80, May 2023.
[49] J. Qi, J. Du, S. M. Siniscalchi, X. Ma, and C. H. Lee, "Analyzing upper bounds on mean absolute errors for deep neural network-based vector-to-vector regression," *IEEE Trans. Signal Process.*, vol. 68, pp. 3411–3422, May 2020.

Jing Bi (Senior Member, IEEE) received the B.S. and Ph.D. degrees in computer science from Northeastern University, Shenyang, China, in 2003 and 2011, respectively.

From 2013 to 2015, she was a Postdoctoral Researcher with the Department of Automation, Tsinghua University, Beijing, China. From 2018 to 2019, she was a Visiting Research Scholar with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. She is a Professor with the Faculty of Information Technology, School of Software Engineering, Beijing University of Technology, Beijing. Her research interests include distributed computing, cloud computing, large-scale data analytics, machine learning, and performance optimization.

Dr. Bi received the IBM Fellowship Award, the Best Paper Award at the 17th IEEE International Conference on Networking, Sensing and Control, and the First-Prize Progress Award of the Chinese Institute of Simulation Science and Technology. She is currently an Associate Editor of IEEE TRANSACTIONS ON SYSTEMS MAN AND CYBERNETICS: SYSTEMS.

Haisen Ma (Student Member, IEEE) received the B.S. degree in software engineering from Zhengzhou University, Zhengzhou, China, in 2021. He is currently pursuing the master's degree with the Faculty of Information Technology, School of Software Engineering, Beijing University of Technology, Beijing, China.

His research interests include cloud computing, data centers, big data, time series prediction, machine learning, and deep learning.

Haitao Yuan (Senior Member, IEEE) received the Ph.D. degree in computer engineering from New Jersey Institute of Technology (NJIT), Newark, NJ, USA, in 2020.

He is an Associate Professor with the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. His research interests include cloud computing, edge computing, data centers, big data, machine learning, deep learning, and optimization algorithms.

Dr. Yuan received the Chinese Government Award for Outstanding Self-Financed Students Abroad, the 2021 Hashimoto Prize from NJIT, and the Best Paper Award in the 17th ICNSC. He serves as an Associate Editor for *Expert Systems with Applications*.

Rajkumar Buyya (Fellow, IEEE) received the B.E degree in computer science and engineering from Mysore University, Mysore, India, in 1992, the M.E. degree in computer science and engineering from Bangalore University, Bengaluru, India, in 1995, and the Ph.D. degree in computer science and software engineering from Monash University, Melbourne, VIC, Australia, in 2002.

He is a Redmond Barry Distinguished Professor and the Director of the Cloud Computing and Distributed Systems Laboratory, The University of Melbourne, Melbourne. He has authored over 850 publications and seven textbooks.

Dr. Buyya is a Highly Cited Computer Science and Software Engineering Author Worldwide, with over 146 900 citations and an H-index of 167. Thomson Reuters recognized him as a "Web of Science Highly Cited Researcher" from 2016 to 2021. He was a Future Fellow of the Australian Research Council from 2012 to 2016.

Jinhong Yang received the Ph.D. degree in computer application technology from Harbin Engineering University, Harbin, Heilongjiang, China, in 2017.

She is a Senior Engineer with CSSC Systems Engineering Research Institute, Beijing, China. Her research interests include machine learning, data mining, knowledge reasoning, deep learning, and intelligent optimization.

Dr. Yang is a reviewer for *Expert Systems With Applications* and *International Journal of Machine Learning and Cybernetics*.

Jia Zhang (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Illinois at Chicago, Chicago, IL, USA, in 2000.

She is the Cruse C. and Marjorie F. Calahan Centennial Chair of Engineering and a Professor of the Department of Computer Science, Lyle School of Engineering, Southern Methodist University, Dallas, TX, USA. Her research interests emphasize applying machine learning and information retrieval methods to tackle data science infrastructure problems, with a recent focus on scientific workflows, provenance mining, software discovery, knowledge graphs, and interdisciplinary applications of all these interests in Earth science.

MengChu Zhou (Fellow, IEEE) received the Ph.D. degree from Rensselaer Polytechnic Institute, Troy, NY, USA, in 1990.

He was then with New Jersey Institute of Technology, Newark, NJ, USA, where he is currently a Distinguished Professor. He has more than 1000 publications, including 12 books, more than 700 journal papers (more than 600 in IEEE Transactions), 31 patents, and 30 book chapters. His interests are Petri nets, automation, Internet of Things, cloud/edge computing, and artificial intelligence.

Dr. Zhou is Fellow of IFAC, AAAS, CAA, and NAI.