Neurocomputing 424 (2021) 35–48

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Integrated deep learning method for workload and resource prediction in cloud systems ☆

Jing Bi [a], Shuang Li [a], Haitao Yuan [b,*], MengChu Zhou [c]

[a] School of Software Engineering in Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China
[b] School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China
[c] Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, USA

ARTICLE INFO

Article history:
Received 12 September 2020
Revised 27 October 2020
Accepted 7 November 2020
Available online 25 November 2020
Communicated by Zidong Wang

Keywords:
Cloud data centers
BG-LSTM
Hybrid prediction
Savitzky–Golay filter
Deep learning

ABSTRACT

Cloud computing providers face several challenges in precisely forecasting large-scale workload and resource time series. Such prediction can help them to achieve intelligent resource allocation for guaranteeing that users' performance needs are strictly met with no waste of computing, network and storage resources. This work applies a logarithmic operation to reduce the standard deviation before smoothing workload and resource sequences. Then, noise interference and extreme points are removed via a powerful filter. A Min–Max scaler is adopted to standardize the data. An integrated method of deep learning for prediction of time series is designed. It incorporates network models including both bi-directional and grid long short-term memory network to achieve high-quality prediction of workload and resource time series. The experimental comparison demonstrates that the prediction accuracy of the proposed method is better than several widely adopted approaches by using datasets of Google cluster trace.

1. Introduction

Nowadays, cloud computing is increasingly built and commonly adopted by many organizations [1]. It establishes a common and dynamic pool of storage, computing and network resources by integrating cloud data center (CDC) networks, servers, software, and storage services. In addition, memory, storage and network bandwidth resources in cloud computing are dynamically allocated according to user needs [2,3]. In 2018, interactive applications accounted for a large percentage of all applications in cloud computing and had 3.6 billion users. Typical cloud providers, *e.g.*, Amazon, Facebook, Google and Alibaba, have built CDCs for users to rent their computing resources [4]. However, the number of users in CDCs is increasing dramatically. Therefore, CDC providers have to manage a large number of user tasks while ensuring their Quality of Services (QoS), increasing providers' profit and reducing their cost.

To provide high resource availability and meet requirements of Service-Level Agreements (SLAs), CDC providers have to conduct proactive resource allocation [5,6]. Allocating resources optimally for workload requires accurate prediction of workload on servers in CDCs. However, it is difficult to achieve so in CDCs because it requires in-depth understanding of characteristics of workload and resource usage in CDCs. Their workload is dynamic and highly fluctuating, and resource usage is constantly changing during the execution of tasks. Currently, most of providers adopt a tiered payment business model that requires users to select types and number of resources, *e.g.*, virtual machine instances, before using their desired services.

General users, especially non-information technology professionals, have no clear understanding of how many resources they may consume. Under normal circumstances, most of them suffer unnecessary cost. It also causes huge waste of resources and reduces revenue of CDC providers. Furthermore, if users select insufficient resources, their tasks may be delayed or even unable to complete. In this way, QoS requirements of users' services cannot be well met. It also will reduce the satisfaction of users and finally lead to the loss of users. If CDC providers can accurately predict the workload and resources that users may need to use in future time slots based on historical workload and resource data,

☆ This work was supported in part by the Major Science and Technology Program for Water Pollution Control and Treatment of China under Grant 2018ZX07111005, in part by the National Natural Science Foundation of China (NSFC) under Grants 62073005 and 61802015, in part by the National Defense Pre-Research Foundation of China under Grants 41401020401 and 41401050102, and in part by the Deanship of Scientific Research (DSR) at King Abdulaziz University, Jeddah, Saudi Arabia, under Grant No. RG-48-135-40.

* Corresponding author.
E-mail address: yuanhaitao@buaa.edu.cn (H. Yuan).

https://doi.org/10.1016/j.neucom.2020.11.011
0925-2312/

they can more effectively manage CDC resources, and obtain larger revenue.

To address this challenging problem, most of current existing studies just consider the prediction of workload time series in CDCs [7]. In this work, the time series prediction of both workload and resource usage in CDCs is comprehensively considered. Currently, there are multiple methods for predicting time series data. For traditional prediction of time series, multiple-output modeling [8], Autoregressive Integrated Moving Average model (ARIMA) [9] and Support Vector Machine (SVM) [10] are some widely used and typical methods. The recent emergence of deep learning models, *e.g.*, Long Short-Term Memory (LSTM) [11] neural networks and Gated Recurrent Unit (GRU) [12], provide new mechanisms to effectively realize high-accuracy time series prediction, and can effectively alleviate the gradient disappearance problem of traditional Recurrent Neural Networks (RNN) [13]. Some studies propose some methods to improve the internal cell of an LSTM network structure and transform its external model structure. Among them, BiLSTM (Bi-directional LSTM) [14] and GridLSTM (Grid LSTM) [15] are two variants that change its external model structures. At present, researchers adopt BiLSTM [17] for cross-language pronoun prediction and GridLSTM [22] for multi-channel speech frequency change prediction. BiLSTM and GridLSTM can capture two directional dependence characteristics and different dimension information, respectively. Due to the highly changing characteristics of workload and resource usage, and the data generated by CDCs during operations increasing with time, traditional prediction methods are insufficient to predict the large-scale data. Their prediction accuracy is unsatisfying. Therefore, different from these studies, our work proposes a novel deep LSTM method that integrates advantages of both BiLSTM and GridLSTM to predict the time series of workload and resource usage, and achieves better prediction performance.

The contributions of this work are:

1. It performs a logarithmic operation to reduce the standard deviation before smoothing workload and resource usage time series. After testing different smoothing methods for time series to exclude noise interference and extreme points, this work identifies the filter of Savitzky-Golay (SG) [23] as the best one among many in terms of the prediction accuracy;
2. It integrates BiLSTM and GridLSTM models, referred to as BG-LSTM, to build a prediction model of workload and resource usage time series. It can effectively extract complex characteristics of such series and achieve high prediction accuracy; and
3. Real-life workload, the Central Processing Unit (CPU) and Random Access Memory (RAM) data are used to conduct experiments for demonstrating that BG-LSTM outperforms many benchmark approaches with respect to the prediction accuracy, particularly for relatively longer time series.

For clarity, we show the differences between our previous yet significantly different studies [7,24] and this work as follows.

1. The study in [24] adopts an SG filter and [7] adopts a wavelet decomposing method to analyze data. This work compares results of various smoothing methods, and identifies the SG filter as the most effective one among them. It presents the method of Min–Max scaler in the data preprocessing, further reducing the scale of original data and increasing prediction accuracy, which are not given in previous studies.
2. In the previous studies, the 29-day Google cluster workload is divided into 8352 time slots, and the time slot length is 5 min. This work adopts the same workload, which is divided into 20880 time slots, and the length of each time slot is 2 min. The obtained workload time series is significantly different from that in the previous studies. It contains much more data samples than that in the previous studies. The previous studies only consider the workload time series. Different from them, this work also considers two types of time series including CPU and RAM resources.
3. Different from previous studies, this work innovatively integrates BiLSTM and GridLSTM models to propose a prediction model called BG-LSTM for both workload and resource usage time series. BiLSTM layers are able to analyze the time series around the current time slot, and they can capture bi-directional dependencies and encode information from reverse time slots into the current one. GridLSTM layers can analyze the time series through the dimension of depth. They can obtain outputs of time and frequency domains, and then concatenate them together. Thus, the proposed BG-LSTM significantly differs from the standard LSTM, and it also has much better generality than that in previous studies.

The literature review is given in Section 2. Section 3 gives the details of the proposed BG-LSTM. Section 4 gives the real-life data-driven experiments. At last, the conclusion is drawn in Section 5.

2. Related work

The continuous deployment of various service ecosystems in CDCs attracts a growing number of users around the world who need services provided by CDCs. Therefore, precisely predicting the trend of tasks and resource usages is becoming increasingly important. There are many methods that can achieve the prediction of time series.

2.1. Classical methods for prediction of time series

Many approaches have been widely used for time series forecasting, *e.g.*, ARIMA [9], Hidden Markov Model (HMM) [18], Support Vector Regression (SVR) [19,20] and Back-Propagation Neural Network (BPNN) [21]. Li *et al.* [22] apply a method of ARIMA to achieve long-term load prediction by considering load variations in some developed cities. Calheiros *et al.* [9] adopt ARIMA method to predict workload in public clouds where workload expresses high fluctuations. However, as a linear model, ARIMA cannot handle non-linearity in workload time series. Similar to ARIMA, a vector auto regression model only investigates linear characteristics in time series data. Therefore, Bao *et al.* [25] design SVR to investigate the non-linear features of multi-variable data. Khan *et al.* [26] propose a workload forecasting model for multiple time series. They adopt a method of HMM to detect the time dependence in virtual machine clusters and predict tendencies of patterns in workload. Baldan et al. [27] combine a neural network of error correction and linear regression models for workload prediction in cloud systems.

Islam *et al.* [28] propose a resource measurement and supply strategy based on prediction, which adopts linear regression models and a neural network based on a sliding window technique to satisfy the upcoming CPU resource need. The evaluation result proves that it outperforms linear regression and neural network models without a technique of sliding window. Lu *et al.* [29] propose a model of workload prediction for a cloud environment based on back propagation learning. Google cluster trace is adopted to evaluate the model by predicting latency-sensitive tasks. Hu *et al.* [30] adopt a statistical learning mechanism to develop a model of prediction that incorporates a Kalman filter [31] and SVR. It achieves higher prediction accuracy than BPNN and SVR. However, if the amount of time series is too large, it is

time or memory-consuming, and its prediction efficiency is unsatisfying. Bi *et al.* [7] design a method of stochastic configuration networks to develop a model for task prediction in clouds. It adopts a wavelet decomposition method to capture detailed information of data at multiple resolutions. However, the proposed method is only tested on a small dataset and fails to consider long-term dependence among data.

2.2. Deep learning methods for time series prediction

Deep learning methods receive dramatic attention along with the rapid evolution of artificial intelligence [32]. Deep learning-based short-term time series prediction is becoming a hot topic in recent years. Compared with traditional shallow learning structures, deep neural networks can investigate the hierarchical and distributed feature expression to analyze complex nonlinear characteristics in data [33]. At present, in addition to conventional methods, deep learning has been diffusely explored in time series prediction [34,35]. Many neural network variants, *e.g.*, stacked auto encoder models [40], Deep Belief Networks (DBN) [41] and LSTM [11] are proposed. Bi *et al.* [24] adopt a simple LSTM method and establish a forecasting model for accurately predicting data center workload. They also use an SG filter to eliminate noise and extreme points in a workload series. However, they only adopt a standard and simple LSTM model and fail to obtain significant gain in prediction accuracy.

Researchers improved internal cells of LSTM and transformed its external model structure, and proposed several variants including GridLSTM [15], Bi-directional LSTM (BiLSTM) [45], and convolutional LSTM [42]. Their key differences lie in connection optimization of LSTM cells to enhance prediction performance of LSTM networks. Wu *et al.* [43] implement a network of deep BiLSTM for prediction of protein intrinsic disorder. The results find its usefulness in investigating long-range and non-local interactions for applications in bioinformatics fields. Danihelka *et al.* [15] find that 2-dimension GridLSTM outperforms stacked LSTM in memorizing digital sequences. The learning properties of GridLSTM can be improved by recurrent connections along a dimension of depth. Barra *et al.* [34] aim to predict the trend of U.S. market, and investigate an ensemble of Convolutional Neural Networks (CNNs), which is trained with images of Gramian angular fields and produced from the time series data about the Standard & Poor's 500 index future. A imaging method of multiple resolutions is adopted to train each CNN, and an effective trading system based on the ensemble predictor is adopted to verify the performance of the method.

Pham *et al.* [35] adopt fuzzy recurrence plots for very short time series data with LSTM networks. This method dramatically increases feature dimensions and gives characteristics of dynamic deterministic systems for very short time series. It applies deep learning and improves the short time series classification for computer-key hold time collected from two cohorts of early Parkinson's disease and healthy control. However, this method only adopts a simple LSTM cell for prediction and fails to use any improved variants of LSTM. Kebria *et al.* [36] evaluate the effect of three architectural parameters including layer count, filter count for CNNs. It focuses on optimization of CNNs that are mainly used to describe the state of space. Different from it, LSTM can focus more on temporal connections among data. Bao *et al.* [37] analyze the memory of a memristor by using mathematical models of memristors with nonlinear and linear drifts. In addition, the analysis of memory in both series and parallel patterns for memristors is also provided. Ouyang *et al.* [38] utilize neural networks to approximate uncertainty values of a system of an uncertain 2-degree of freedom helicopter, and dead-zone nonlinearity for controlling the system. However, this study mainly focuses on using standard neural networks to generate a solution for coping with system uncertainties and input dead-zone. The work [39] performs bradykinesia recognition in parkinsons disease by using video data series by using onvolution neural networks and an encoder--decoder structure. Different from these studies, this work integrates GridLSTM and BiLSTM models to extract general characteristics of workload and resource usage time series, and adopts an SG filter to smooth such series from CDCs to eliminate outliers and noise.

3. Model framework

3.1. LSTM

To reveal long-term dependence of time series prediction, Hochreiter *et al.* [11] propose LSTM to analyze time series data. The prediction of a workload time series implies capturing and using past changing patterns from the past workload series. LSTM improves the memory ability of traditional RNN with a memory cell. Fig. 1 shows a standard LSTM cell. Each cell contains a number of gate units, which can determine input data, cell status and output. The standard cell is mathematically described as:

$$\begin{aligned} f_t &= \sigma(W_{fh}h_{t-1} + W_{fx}x_t + b_f) \\ i_t &= \sigma(W_{ih}h_{t-1} + W_{ix}x_t + b_i) \\ \tilde{c}_t &= \tanh\left(W_{\tilde{c}h}h_{t-1} + W_{\tilde{c}x}x_t + b_{\tilde{c}}\right) \\ c_t &= f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t \\ o_t &= \sigma(W_{oh}h_{t-1} + W_{ox}x_t + b_o) \\ h_t &= o_t \cdot \tanh(c_t) \end{aligned} \tag{1}$$

where f, i and o represent outputs of three gate units, *i.e.*, forget, input and output ones. c means the state of the LSTM cell. x denotes the input of a cell. b and W are bias and weight matrix of the cell, which are parameters for its corresponding gates and cell state. h denotes the recurrent information among cells. t and $t-1$ denote current and previous time instances. $tanh(\cdot)$ and $\sigma(\cdot)$ mean hyperbolic and sigmoid activation functions. The operator $\cdot$ means the point-wise multiplication of given two vectors.

Different from feed forward neural networks, LSTM can process a sequence of any length, and can handle inputs and outputs with variable length. It has the ability of storing long-term memory and processing contextual information. In addition, LSTM can solve the problems of vanishing and exploding gradient which may be encountered in RNN. It also has the ability to quickly adapt to sharp changes in the trend, and owns better processing ability in the prediction of volatile time series.

3.2. BG-LSTM

To seek high prediction accuracy, three methods are used in data preprocessing. An SG filter is first adopted to decrease the noise information in the original data including workload and resource usage. Then, the natural logarithm [45] and Min–Max scaler [46] are adopted to reduce the scale of the original data. After such data preprocessing, BiLSTM and GridLSTM are integrated to train and test time series data. Fig. 2 shows a specific model prediction structure. The input of the proposed prediction model is generated from the preprocessed data. The model includes a GridLSTM layer, which is in the middle of two BiLSTM layers. After that, the output of BiLSTM goes through a fully connected layer for yielding the output finally.

In reality, many time series have strong non-stationarity and high heterogeneity. A single prediction model is difficult to capture all features of time series of original data, and it is difficult to develop a model for precise prediction. Some recent studies have

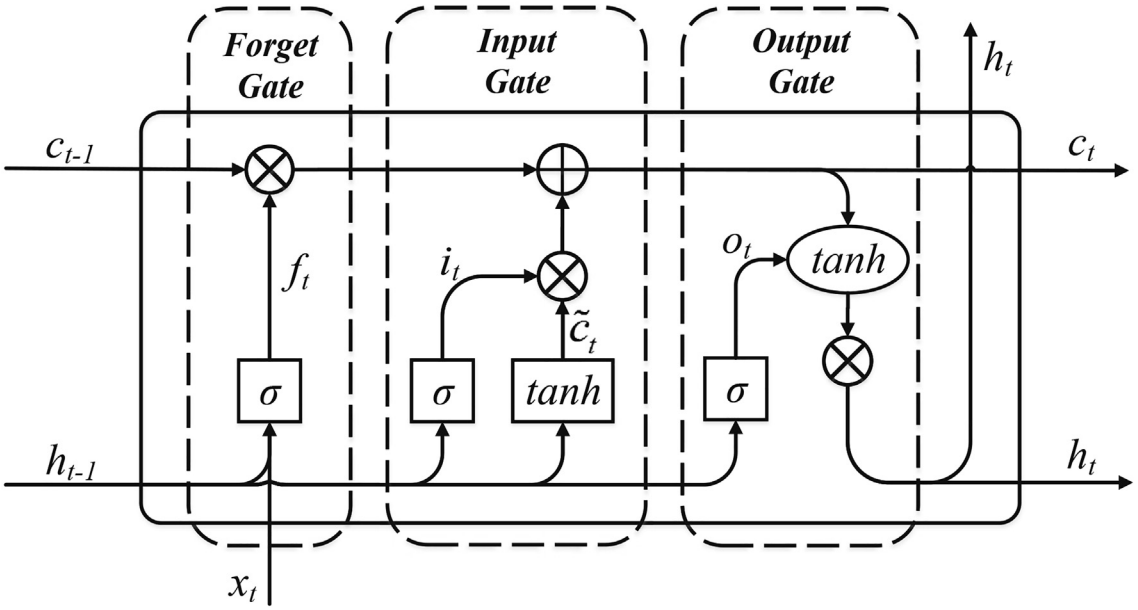


Fig. 1. A cell of LSTM.

been proposed on integration and combination of different models. Fei *et al.* [39] develop an approach that considers gradient and context-sensitivity problems. They construct an improved bidirectional structure by using GridLSTM. Integrating different prediction methods can extract multi-dimension features of the original data, and improve the prediction of time series. Thus, the prediction accuracy of the original complex time series is improved.

Different from [39], to obtain high-accuracy prediction by considering features of data, this work combines BiLSTM [14] and GridLSTM [15] to extract potential features of time series data in CDCs. BiLSTM layers are able to explicitly model the time series around current time slot while GridLSTM layers can model the time series by using the dimension of depth. To extract deeper features of the time series data, this work adopts a two-layer BiLSTM that has strong ability to produce comprehensive data analysis [16]. The flowchart of the proposed prediction model based on BG-LSTM is given in Fig. 2. The specific steps of implementation are given as follows.

1. First, this work collects the historical data in Google cluster trace. By analyzing and organizing the timestamp information of tasks, the numbers of tasks and resource usage records including CPU and RAM, are counted for each time slot. Then, workload and resource usage time series are obtained as historical time series used in the experiments.
2. Second, three methods are adopted in the data preprocessing. Due to the enormous scale of raw workload and resource usage time series, the natural logarithm [45] is first adopted to reduce the scale of the original data. Besides, an SG filter is adopted to decrease the noise information in the original data. Then, the Min–Max scaler [46] is adopted to keep each feature of data in the same order of magnitude.
3. Third, after such data preprocessing, BiLSTM and GridLSTM are integrated together to train and test data of time series. The input of BG-LSTM is generated from the preprocessed data. BG-LSTM includes a GridLSTM layer, which is in the middle of two BiLSTM layers. After that, the output of BiLSTM goes through a fully connected layer for yielding the final output.

The output of BG-LSTM is defined as follows. BiLSTM and GridLSTM are improved models of LSTM, and the structure that is similar to LSTM replaced with $\mathbb{L}(\cdot)$:

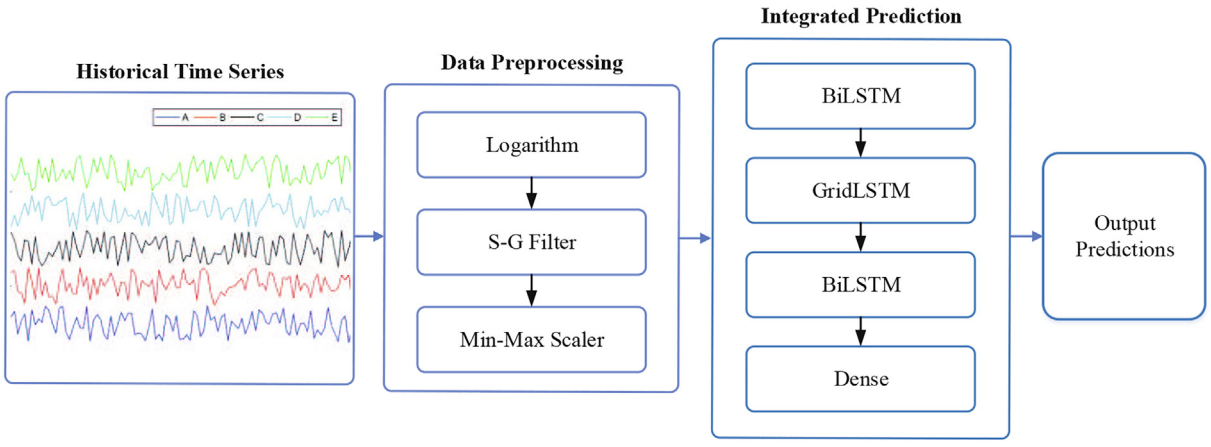
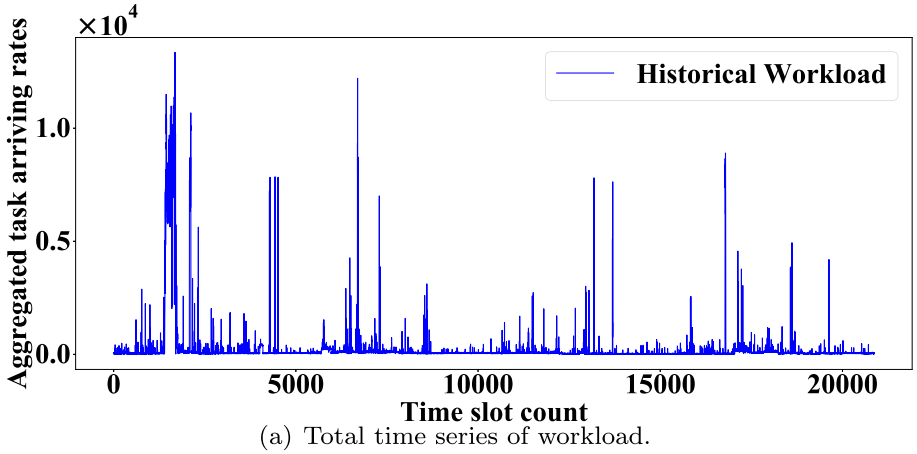


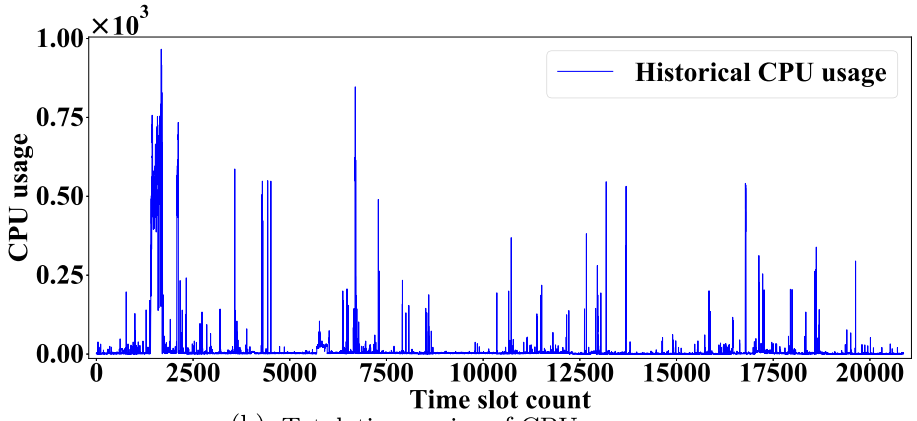
Fig. 2. Data preprocessing and BG-LSTM structure.

$$
\begin{aligned}
\overleftarrow{O}_t^L &= \mathbb{L}\left(\overleftarrow{f}_t^L, \overleftarrow{i}_t^L, \overleftarrow{o}_t^L, \overleftarrow{h}_{t-1}^L, I_t\right) \\
\overrightarrow{O}_t^L &= \mathbb{L}\left(\overrightarrow{f}_t^L, \overrightarrow{i}_t^L, \overrightarrow{o}_t^L, \overrightarrow{h}_{t-1}^L, I_t\right) \\
O_t^{L+1} &= \mathbb{L}\left(f_t^{L+1}, i_t^{L+1}, o_t^{L+1}, h_{t-1}^{L+1}, O_t^L\right) \\
\overleftarrow{O}_{t+1}^{L+1} &= \mathbb{L}\left(\overleftarrow{f}_{t+1}^{L+1}, \overleftarrow{i}_{t+1}^{L+1}, \overleftarrow{o}_{t+1}^{L+1}, \overleftarrow{h}_t^{L+1}, O_t^{L+1}\right) \\
\overrightarrow{O}_{t+1}^{L+1} &= \mathbb{L}\left(\overrightarrow{f}_{t+1}^{L+1}, \overrightarrow{i}_{t+1}^{L+1}, \overrightarrow{o}_{t+1}^{L+1}, \overrightarrow{h}_t^{L+1}, O_t^{L+1}\right) \\
y_{t+1} &= W_{\overrightarrow{h}y}\overrightarrow{O}_{t+1}^{L+1} + W_{\overleftarrow{h}y}\overleftarrow{O}_{t+1}^{L+1} + b_y
\end{aligned}
\tag{2}
$$

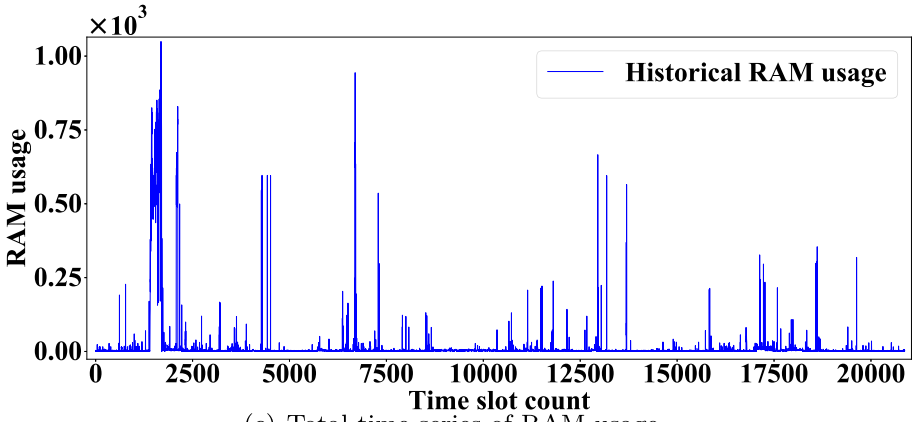
where f, i and o denote outputs of three gate units, which are forget, input and output gates, respectively. I denotes an input of BG-LSTM. $\overleftarrow{O}_t^L$ and $\overrightarrow{O}_t^L$ denote outputs of BiLSTM layer L. $\overleftarrow{O}_{t+1}^{L+1}$ and $\overrightarrow{O}_{t+1}^{L+1}$ denote outputs of BiLSTM layer $L+1$. O_t^{L+1} denotes an output of a GridLSTM layer. b and W are the bias and weight matrix of gate units. h and t denote recurrent information and time instance among models, respectively. The superscript $\rightarrow$ denotes the sequence from $t=1$ to T, and $\leftarrow$ denotes the sequence from $t=T$ to 1. y_{t+1} represents outputs of BG-LSTM.



(a) Total time series of workload.

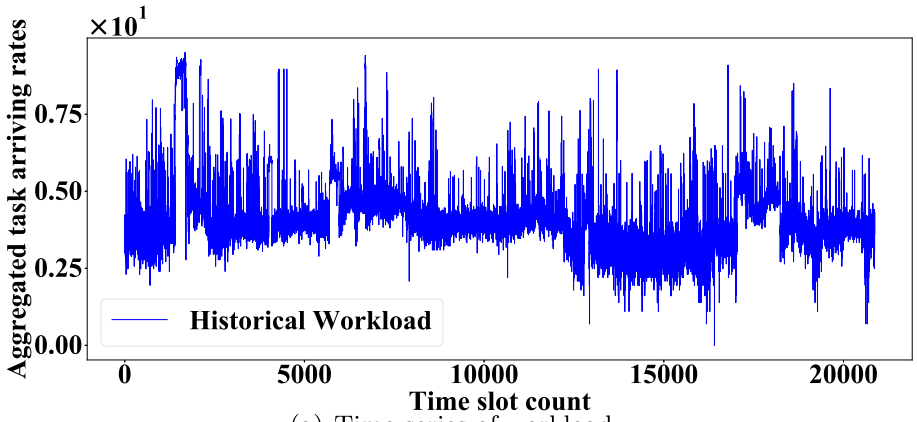


(b) Total time series of CPU usage.

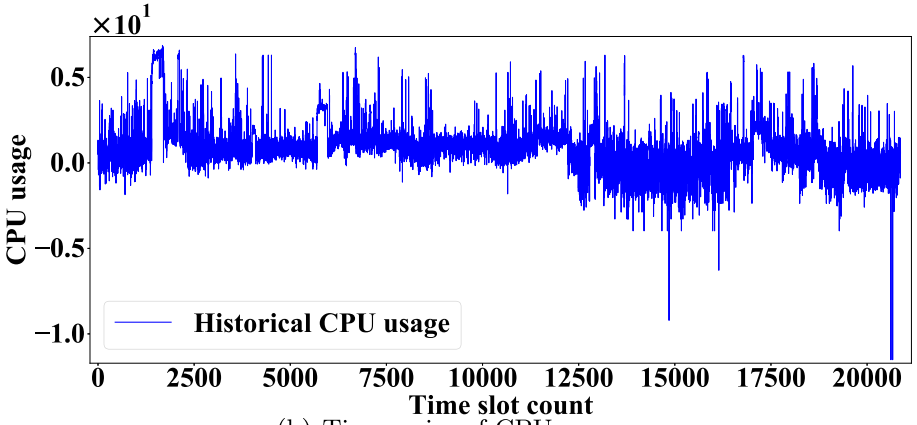


(c) Total time series of RAM usage.

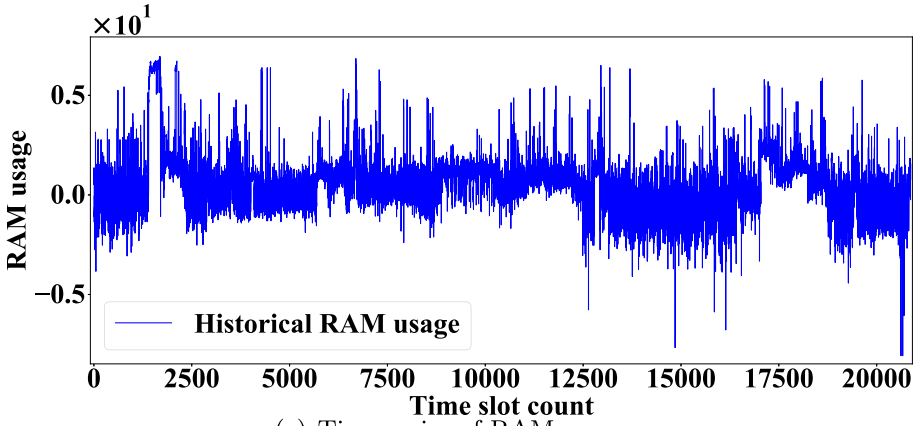
Fig. 3. Total workload and resource time series from Google clusters.



(a) Time series of workload.



(b) Time series of CPU usage.

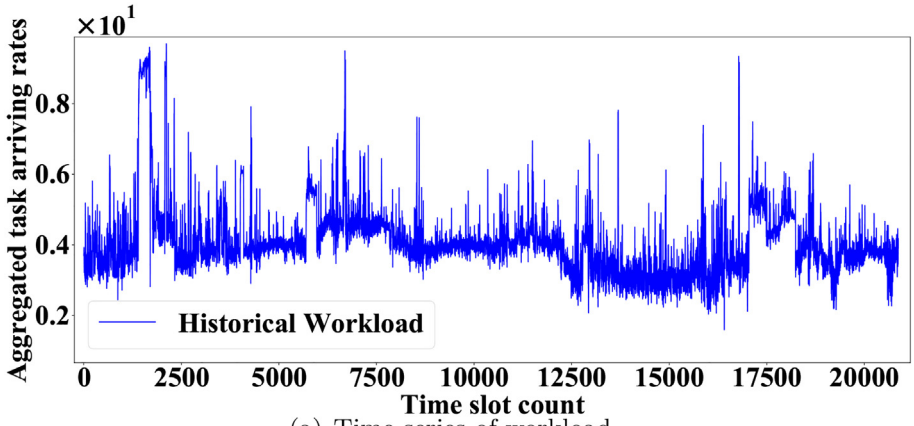


(c) Time series of RAM usage.

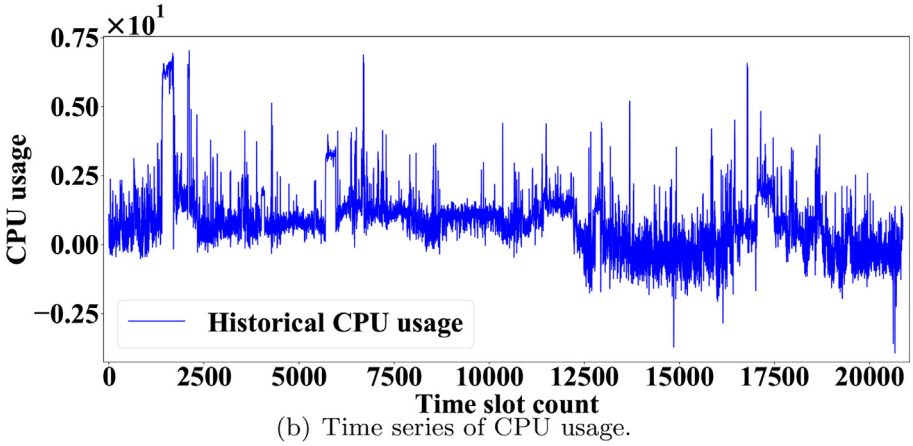
Fig. 4. Total workload and resource time series data after taking natural logarithm.

Table 1
Performance comparison of different filters.

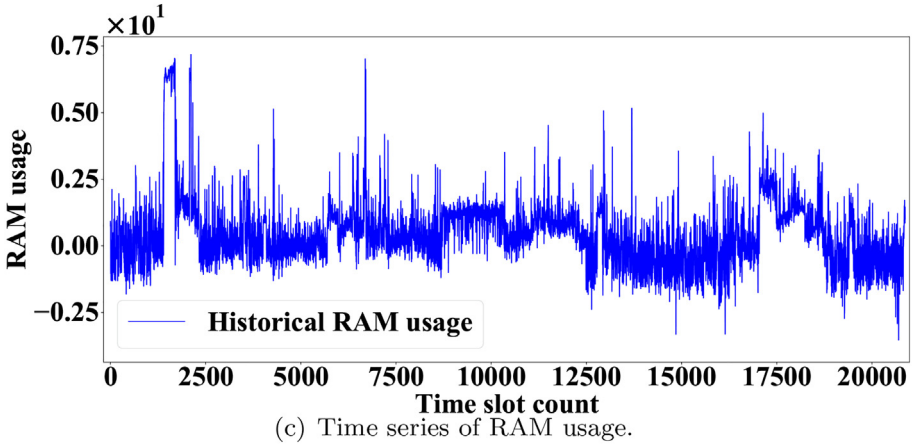
Methods	Workload			CPU			RAM		
	RMSLE	MSE	R^2	RMSLE	MSE	R^2	RMSLE	MSE	R^2
No filter	0.72	61631.36	0.91	0.78	416.28	0.83	0.80	531.25	0.88
Median filter	0.46	40782.84	0.94	0.57	284.68	0.91	0.55	302.91	0.92
Average filter	0.18	15988.27	0.97	0.23	162.34	0.98	0.21	189.71	0.95
SG filter	**0.15**	**13934.54**	**0.99**	**0.16**	**128.89**	**0.99**	**0.14**	**131.29**	**0.99**



(a) Time series of workload.



(b) Time series of CPU usage.



(c) Time series of RAM usage.

Fig. 5. Total workload and resource time series data with the SG filter.

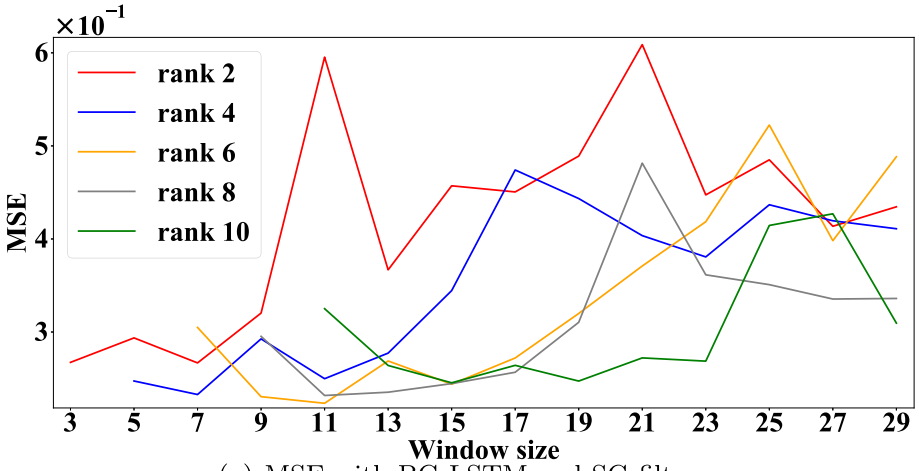
4. Performance evaluation

This section evaluates the proposed method by testing and verifying its efficiency and usability with real-world datasets as well as its comparison with its widely used peers.

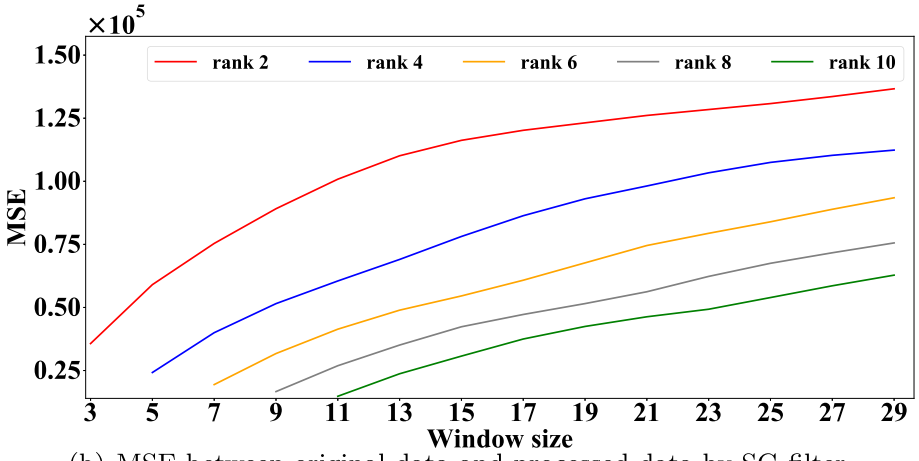
4.1. Preprocessing of data

To investigate characteristics of actual task and resource usage data, this section chooses the workload and resource trace collected from compute clusters in Google.[1] The Google cluster data trace was released in 2011 [47], and many recent studies apply it to evaluate the prediction performance of their proposed methods [48–51]. It has become a de facto and widely adopted data set by many newly emerging studies. Therefore, similar to them, this work also adopts this data set to evaluate the prediction performance of BG-LSTM. The data contains about 12,000 computers. This workload data contains 25,462,157 tasks and 672,003 jobs in 29 days. Workload arrives as jobs, each of which has multiple tasks. Each task can be viewed as a Linux program that needs to run on a specific computer. Our work develops the model of BG-LSTM to predict workload and resource usage time series. This work divides the total length of 29 days into 20880 time slots of each length. Each time slot lasts for 2 min. By analyzing the timestamp information of tasks, the

[1] https://github.com/google/cluster-data.



(a) MSE with BG-LSTM and SG filter.



(b) MSE between original data and processed data by SG filter.

Fig. 6. MSE with respect to different window sizes and rank values.

Table 2
Training process of *X*, *Batch_size* and *Epochs* using workload time series.

X	*Batch_size*&*Epochs* = **5000**&**40000**		*Batch_size*&*Epochs* = 2000&20000	
	RMSLE	Training time (s)	RMSLE	Training time (s)
10	0.28	4721.36	0.30	4656.81
20	0.21	4753.93	0.21	4819.71
30	0.19	5438.59	0.22	4356.33
40	0.18	4700.98	0.18	4528.34
50	0.18	4507.07	0.17	5072.99
60	**0.15**	4732.95	0.24	4294.47
70	0.17	5100.70	0.25	4419.85
80	0.19	5010.87	0.18	4492.84
90	0.18	5519.07	0.23	4505.10
100	0.21	6110.29	0.18	4662.58
150	0.21	5580.20	0.20	4843.53
200	0.24	6169.42	0.23	5041.25
X	*Batch_size*&*Epochs* = 1000&10000		*Batch_size*&*Epochs* = 500&5000	
	RMSLE	Training time (s)	RMSLE	Training time (s)
10	0.24	4294.47	0.28	3838.34
20	0.21	4235.90	0.22	3870.37
30	0.20	5066.33	0.19	3913.14
40	0.18	4239.23	0.19	3934.48
50	0.18	4260.88	0.19	3968.03
60	0.17	3983.82	0.18	4434.75
70	0.17	4023.61	0.21	5042.39
80	0.19	4782.22	0.18	4059.77
90	0.17	4866.10	0.20	4251.14
100	0.18	5652.15	0.20	5766.94
150	0.19	5378.23	0.21	6158.26
200	0.23	6004.97	0.24	6852.14

Table 3
Training process of *X*, *Batch_size* and *Epochs* using resources time series.

X	*Batch_size*&*Epochs* = **5000**&**40000**		*Batch_size*&*Epochs* = 2000&20000	
	RMSLE	Training time (s)	RMSLE	Training time (s)
10	0.36	5728.86	0.32	4119.36
20	0.29	5021.88	0.29	4417.80
30	0.19	6208.24	0.20	4939.29
40	0.37	6153.80	0.19	4684.94
50	0.17	5168.51	0.21	4685.03
60	**0.16**	5163.46	0.17	4660.60
70	0.25	5346.09	0.18	4833.57
80	0.19	5452.87	0.18	4700.51
90	0.20	5521.35	0.22	4715.59
100	0.19	5593.49	0.18	4862.58
150	0.21	5716.32	0.20	5033.71
200	0.22	5869.33	0.23	5249.55
X	*Batch_size*&*Epochs* = 1000&10000		*Batch_size*&*Epochs* = 500&5000	
	RMSLE	Training time (s)	RMSLE	Training time (s)
10	0.34	4221.32	0.28	4232.80
20	0.27	4231.76	0.23	3874.82
30	0.21	4326.51	0.19	4004.32
40	0.18	4341.43	0.18	4024.68
50	0.18	4260.88	0.17	3954.04
60	0.18	4383.82	0.18	4192.45
70	0.17	4023.61	0.18	4230.05
80	0.18	4782.22	0.24	4084.64
90	0.21	4866.10	0.25	4200.38
100	0.18	4852.63	0.20	4268.60
150	0.23	4771.43	0.21	4397.43
200	0.23	5001.04	0.24	4154.18

number of tasks and that of resources including CPU and RAM usage records are counted. Finally, workload and resource usage time series are obtained. It is worth noting that, as shown in Fig. 3, the workload and resource time series mean the total task trace and resource usage data collected from Google clusters for 29 days.

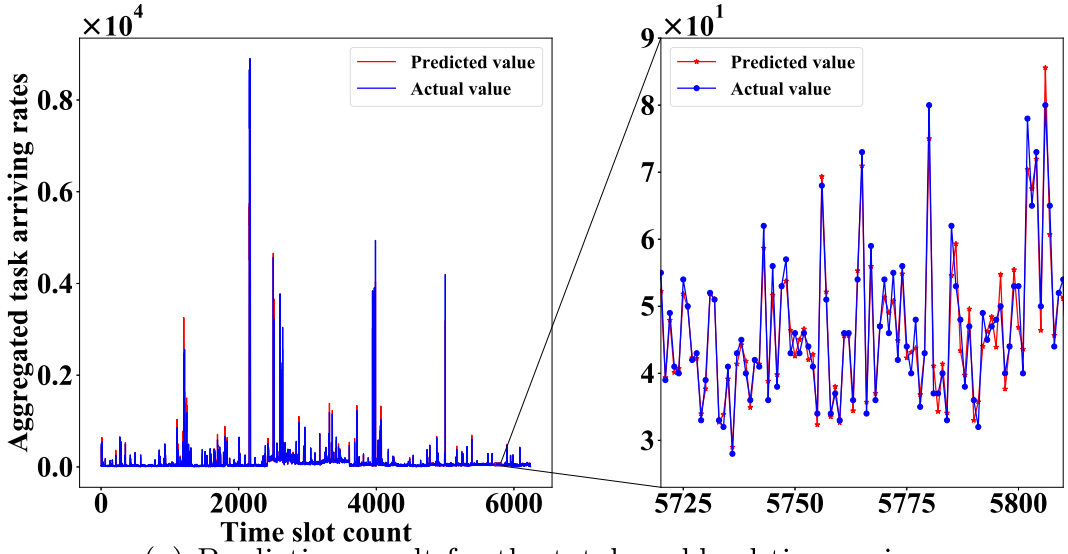
However, original workload and resource usage time series contain much noise caused by physical machine failures in CDCs or other abnormal cases, *e.g.*, the number of abnormal workload and resource usage caused by some unusual activities. This means the workload and resource usage time series contain many extreme points that are much larger than others. If they are not excluded from data when a prediction model is built, the prediction accuracy of data would be seriously damaged. In addition, the distribution of a majority of workload and resource usage time series is very uneven. Most of these data are between 10 and 1000, but there are still much data between 1000 and 10,000. Therefore, the workload and resource usage data are highly non-linear, non-stationary, and varying greatly, and therefore they are complicated to achieve the accurate prediction. Here, this section adopts the natural logarithm before smoothing such that the magnitude of total workload and resource usage time series is greatly reduced. The processed series become easier to build a more accurate model for prediction than the original time series.

As shown in Fig. 4, this section takes the natural logarithm of the original workload and resource time series to reduce the standard deviation. We further compare several filter algorithms to filter outliers and noise. This section selects three evaluation metrics, *i.e.*, Root Mean Squared Logarithmic Error (RMSLE) [52], Mean Square Error (MSE) and R^2 [55]. The prediction results with different filters under the same structure of BG-LSTM are shown in Table 1, and it is observed that the SG filter has better performance than widely used median and average filters. As shown in Fig. 5, the processed workload and resource data are smoothed and processed by the SG filter to exclude potential noise and outliers. In our method, the SG filter with the rank of 6 and the window size of 11 is finally selected.

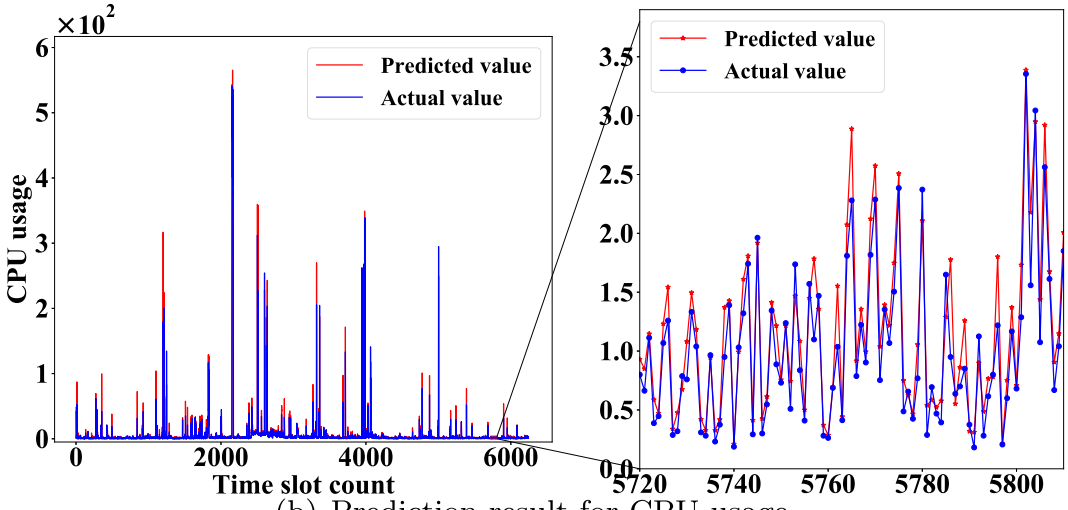
Fig. 6(a) shows the MSE [53] between the original data and predicted data with BG-LSTM and the SG filter with respect to different window sizes and rank values. It is shown in Fig. 6(a) that after multiple trials, the SG filter achieves the minimum change of the original shape of the data when the window size and the rank are set to 11 and 6, respectively. Therefore, the SG filter with such parameter setting is adopted to establish the model. Fig. 6(b) illustrates the MSE between the data processed by the SG filter and the original one. In Fig. 6(b), MSE is adopted to calculate the amount of change between the smoothed data and original one. Larger MSE means a greater change between the smoothed data and original one. According to results shown in Fig. 6(b), the SG filter with the rank of 6 and the window size of 11 is finally selected in our method.

4.2. BG-LSTM

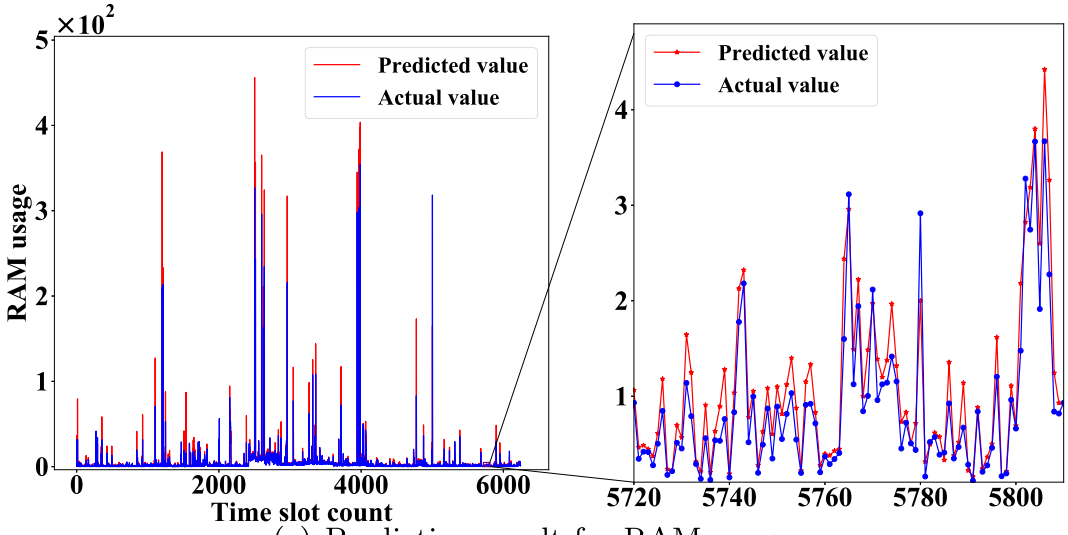
We find that workload and resource usage time series have highly non-stationary and non-linear characteristics with dramatic disturbance. The preprocessed data is then separated into three sections. We adopt the Google dataset as an example. The first 16-day data is selected for training. Data in the middle 4 days is selected for verification. Data in the last 9 days is used as the testing set. To present BG-LSTM with the preprocessing of data, we reorganize the original data to generate the input data. In Fig. 2, we further provide the input data to an integrated model of prediction, which includes four layers: two BiLSTM layers, a GridLSTM one and a dense one. Similar to [54], the best combination of parameters of BG-LSTM is systemically investigated by conducting multiple trials and experiments. Specifically, Tables 4 and 5 show the setting of BG-LSTM parameters for workload and resource time series. We choose *X*, *Batch_size* and *Epochs* to conduct many experiments to select the best setting of parameters because these three parameters are dynamic, important and adjustable during the model training stage. In addition, *Batch_size* and *Epochs* are interrelated parameters. When *Batch_size* is too large and *Epochs* is too



(a) Prediction result for the total workload time series.



(b) Prediction result for CPU usage.



(c) Prediction result for RAM usage.

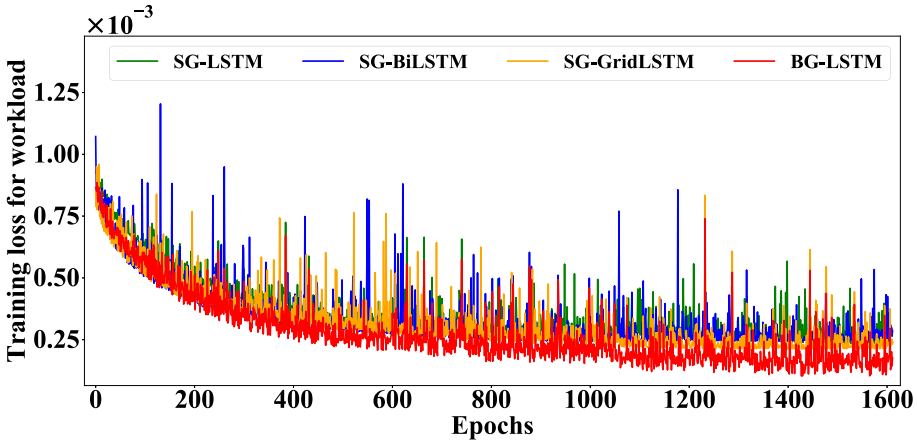
Fig. 7. Prediction results for three datasets with BG-LSTM.

Table 4
Parameter setting of BG-LSTM for workload.

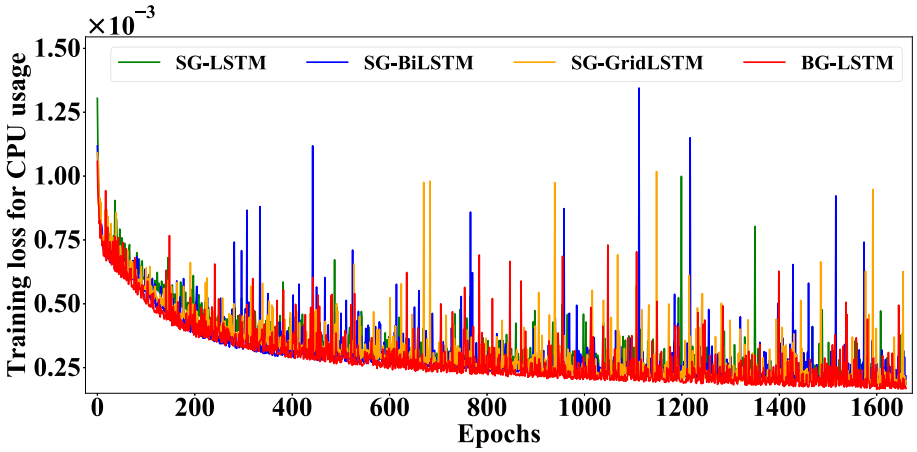
Parameter	Value	Meaning
Y	1	Output of network
X	60	Input of network
Optimizer	Adam	Optimization objective function
Structure	60,45,30,15,1	Structure of network
Epochs	40000	Number of iterations
Batch_size	5000	Size of each batch

Table 5
Parameter setting of BG-LSTM for resources.

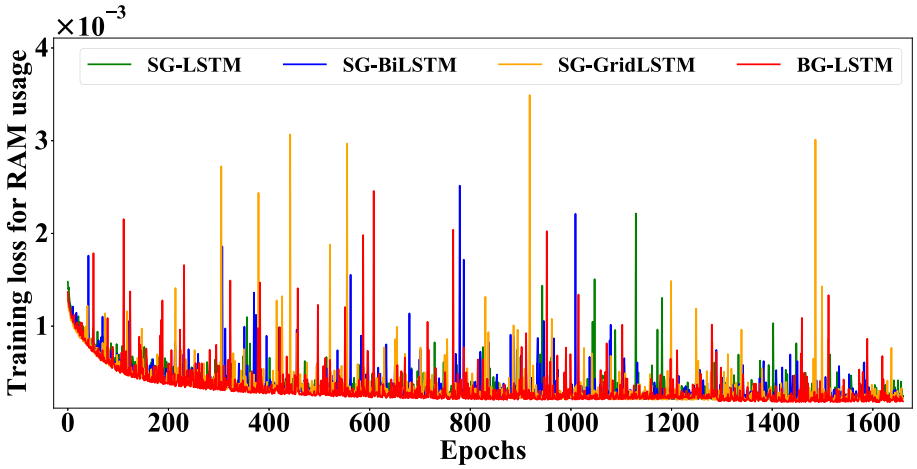
Parameter	Value	Meaning
Y	1	Output of network
X	60	Input of network
Optimizer	Adam	Optimization objective function
Structure	60,50,45,20,1	Structure of network
Epochs	40000	Number of iterations
Batch_size	4000	Size of each batch



(a) Training loss of workload.



(b) Training loss of CPU usage.



(c) Training loss of RAM usage.

Fig. 8. Training loss (Take one every 25 data points).

Table 6
Performance comparison of multiple approaches.

Approaches	Workload			CPU			RAM		
	RMSLE	MSE	R^2	RMSLE	MSE	R^2	RMSLE	MSE	R^2
ARIMA	0.93	107657.34	0.69	0.77	531.34	0.83	0.81	716.52	0.84
SVM	0.86	98133.74	0.71	0.67	410.25	0.87	0.78	601.74	0.86
LSTM	0.83	86753.92	0.69	0.56	313.01	0.91	0.61	351.15	0.92
BiLSTM	0.77	88643.62	0.76	0.58	330.09	0.91	0.69	420.46	0.87
GridLSTM	0.80	82407.57	0.70	0.63	307.58	0.93	0.75	549.26	0.90
SG-LSTM	0.74	73116.80	0.71	0.23	243.62	0.93	0.22	231.99	0.94
SG-BiLSTM	0.19	28995.24	0.97	0.19	182.53	0.95	0.15	151.07	0.96
SG-GridLSTM	0.17	34251.78	0.96	0.20	192.28	0.97	0.16	140.45	0.97
BG-LSTM	**0.15**	**13934.54**	**0.99**	**0.16**	**128.89**	**0.99**	**0.14**	**131.29**	**0.99**

small, it is hard to capture the features of data. Otherwise, it is easy to cause over-fitting. Therefore, *Batch_size* and *Epochs* need to be increasing or decreasing at the same time. The training results of three parameters for workload and resource time series are shown in Tables 2 and 3, respectively. This work sets the values of parameters by first considering RMSLE. The time of training is the second important metric in the parameter setting.

4.3. Prediction results

Fig. 7 shows the results of prediction on three different datasets with BG-LSTM. Fig. 7(a) shows the prediction result of the actual workload. Clearly, the proposed model achieves high-accuracy prediction of the total workload time series. Fig. 7(c) show the prediction results for CPU and RAM usage data, respectively.

To prove the effectiveness and robustness of BG-LSTM, several simulations are conducted repeatedly on the random data from the workload and resource usage data, which is illustrated in Table 6. The three evaluation metrics are MSE, RMSLE and R^2 [55]. They include traditional methods, *e.g.*, ARIMA and SVM, and deep learning methods, *e.g.*, LSTM, BiLSTM, GridLSTM, SG-LSTM, SG-BiLSTM, and SG-GridLSTM. Here SG- means that before using the model to predict data, and the SG filter is first used to process the data. It is observed from Table 6 that the deep learning methods perform better than the traditional ones. In addition, after applying the SG filter method, RMSLE of each method is significantly improved. It is observed BG-LSTM in current work achieves high-accuracy prediction of the total workload time series. Among them, the BG-LSTM that combines BiLSTM and GridLSTM performs the best in terms of RMSLE. Besides, BG-LSTM outperforms SG-LSTM in [24] in terms of all three evaluation metrics for workload and resource time series. The values of three metrics including RMSLE, MSE and R^2 of BG-LSTM (SG-LSTM) are 0.15 (13934.54), 0.99 (73116.80), 0.23 (0.71) for workload time series, respectively. In addition, BG-LSTM yields similar high-accuracy prediction results for resource time series. The comparison results demonstrate that BG-LSTM has better modeling ability than SG-LSTM because it can capture bi-directional dependencies, time and frequency domain features of these time series. Therefore, BG-LSTM improves the fitting ability of SG-LSTM, and it outperforms several typical improved LSTMs with the same parameter setting.

Besides, BG-LSTM performs better than SG-LSTM proposed in [29] in terms of all three evaluation metrics for workload and resource time series. The values of three metrics including RMSLE, MSE and R^2 of BG-LSTM (SG-LSTM) are 0.15 (13934.54), 0.99 (73116.80), 0.23 (0.71) for workload time series, respectively. In addition, BG-LSTM yields similar high-accuracy prediction results for resource time series. The comparison results demonstrate that BG-LSTM has better modeling ability than SG-LSTM because it can capture bi-directional dependencies, time and frequency domain features of these time series. Therefore, BG-LSTM improves the fitting ability of SG-LSTM, and it outperforms several typical improved LSTMs given the same setting of parameters.

Fig. 8(c) compare the training losses of SG-LSTM, SG-BiLSTM, SG-GridLSTM and BG-LSTM models for workload and resource time series, respectively. The training loss of BG-LSTM is lower than those of SG-LSTM, SG-BiLSTM, and SG-GridLSTM when the epoch number reaches 6250 and 11250 for workload and resources, respectively. The training loss is reducing with the increase of epochs, and it is quicker than SG-LSTM, SG-BiLSTM and SG-GridLSTM. The combination of BiLSTM and GridLSTM layers in BG-LSTM has greater modeling capacity than standard LSTM or improved variants of LSTM alone when they are applied to workload and resource time series. BiLSTM layers are able to explicitly model the time series around current time slot, and they can capture bi-directional dependencies and encode information from future time slots to previous ones. Moreover, GridLSTM layers can model the time series through the dimension of depth. They can obtain outputs of time and frequency domains, and then concatenate them together. Thus, these two models complement implicit modeling ability of LSTMs, and therefore BG-LSTM outperforms several typical improved variants of LSTMs with the same parameter setting.

5. Conclusion

Accurate prediction of complex and varying workload and resource usage series is critically important in the efficient operations of cloud data centers (CDCs). Because of their complicated characteristics, to precisely predict them is a major challenge. This work adopts a Savitzky-Golay filter to make the resource and workload data smooth, thus resulting easier-to-predict time series. Then, this work proposes an integrated prediction model, named BG-LSTM, which is composed of a Bi-directional Long Short-Term Memory (LSTM) model and a Grid LSTM one. BG-LSTM is then adopted to investigate characteristics in the workload and resource usage data, and improve the prediction precision in CDCs. Finally, real-world datasets are adopted to prove that BG-LSTM significantly outperforms several widely used traditional deep learning models.

Our future work aims to deal with the sparse and high-dimensional input data with the recently proposed methods [56–58]. In addition, although the current BG-LSTM performs well on the Google dataset, we should further improve it *via* intelligent optimization [59,60] and apply it to more real-life prediction problems in the future. Then, its robustness can be further evaluated by using more types of datasets.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Jing Bi: Methodology, Software, Validation, Formal analysis, Writing - original draft. **Shuang Li:** Software, Validation, Visualization. **Haitao Yuan:** Conceptualization, Formal analysis, Writing - review & editing. **MengChu Zhou:** Validation, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Y. Xia, M. Zhou, X. Luo, Q. Zhu, J. Li, Y. Huang, Stochastic modeling and quality evaluation of infrastructure-as-a-service clouds, IEEE Transactions on Automation Science and Engineering 12 (1) (Jan. 2015) 162–170.

[2] H. Jin, X. Wang, S. Wu, S. Di, X. Shi, Towards optimized fine-grained pricing of iaas cloud platform, IEEE Transactions on Cloud Computing 3 (4) (2015) 436–448.

[3] N. Kumar, G.S. Aujla, S. Garg, K. Kaur, R. Ranjan, S.K. Garg, Renewable energy-based multi-indexed job classification and container management scheme for sustainability of cloud data centers, IEEE Transactions on Industrial Informatics 15 (5) (2019) 2947–2957.

[4] H. Yuan, J. Bi, M. Zhou, Multiqueue scheduling of heterogeneous tasks with bounded response time in hybrid green iaas clouds, IEEE Transactions on Industrial Informatics 15(10) (2019) 5404–5412.

[5] K. Kaur, S. Garg, G. Kaddoum, E. Bou-Harb, K.R. Choo, A Big data-enabled consolidated framework for energy efficient software defined data centers in IoT setups, IEEE Transactions on Industrial Informatics 16(4) (2020) 2687–2697.

[6] H. Yuan, J. Bi, W. Tan, M. Zhou, B.H. Li, J. Li, TTSA: an effective scheduling approach for delay bounded tasks in hybrid clouds, IEEE Transactions on Cybernetics 47(11) (2017) 3658–3668.

[7] J. Bi, H. Yuan, M. Zhou, Temporal prediction of multiapplication consolidated workloads in distributed clouds, IEEE Transactions on Automation Science and Engineering 16 (4) (2019) 1763–1773.

[8] S.B. Taieb, A. Sorjamaa, G. Bontempi, Multiple-output modeling for multi-step-ahead time series forecasting, Neurocomputing 73 (10–12) (2010) 1950–1957.

[9] R.N. Calheiros, E. Masoumi, R. Ranjan, R. Buyya, Workload prediction using ARIMA model and its impact on cloud applications QoS, IEEE Transactions on Cloud Computing 3 (4) (2015) 449–458.

[10] L. Cao, Support vector machines experts for time series forecasting, Neurocomputing 51 (Apr. 2003) 321–339.

[11] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (Mar. 1997) 1735–1780.

[12] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014, arXiv:1412.3555.

[13] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, Nature 323 (6088) (1986) 533–536.

[14] J. Wang, A. Yuille, Semantic part segmentation using compositional model combining shape and appearance, in: Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1788–1797.

[15] I. Danihelka, N. Kalchbrenner, A. Graves, Grid long short-term memory, 2016. arXiv:1507.01526.

[16] Y. Nie, M. Bansal, Shortcut-stacked sentence encoders for multi-domain inference, 2017. arXiv:1708.02312.

[17] S. Stymne, S. Loáiciga, F. Cap, A BiLSTM-based system for cross-lingual pronoun prediction, in: Proc. of the 3rd workshop on discourse in machine translation, Copenhagen, Denmark, 2017, pp. 47–53.

[18] C. Fuh, A.G. Tartakovsky, Asymptotic bayesian theory of quickest change detection for hidden markov models, IEEE Transactions on Information Theory 65 (1) (2019) 511–529.

[19] L. Zhang, W.D. Zhou, P.C. Chang, J.W. Yang, F.Z. Li, Iterated time series prediction with multiple support vector regression models, Neurocomputing 99 (2013) 411–422.

[20] B. Gu, V.S. Sheng, K.Y. Tay, W. Romano, S. Li, Incremental support vector learning for ordinal regression, IEEE Transactions on Neural Networks & Learning Systems 26 (7) (2015) 1403–1416.

[21] J. Kumar, A.K. Singh, Workload prediction in cloud using artificial neural network and adaptive differential evolution, Future Generation Computer Systems 81 (2018) 41–52.

[22] B. Li, T. Sainath, A. Narayanan, J. Caroselli, M. Bacchiani, A. Misra, I. Shafran, H. Sak, G. Pundak, K. Chin, K. Sim, R. Weiss, K. Wilson, E. Variani, C. Kim, O. Siohan, M. Weintraub, E. McDermott, R. Rose, M. Shannon, Acoustic modeling for google home, in: Proc. 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, 2017, pp. 86–91.

[23] A. Savitzky, M.J.E. Golay, Smoothing and differentiation of data by simplified least squares procedures, Analytical Chemistry 36 (1964) 1627–1639.

[24] J. Bi, S. Li, H. Yuan, Z. Zhao, H. Liu, Deep neural networks for predicting task time series in cloud computing systems, in: Proc. 2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC), Banff, AB, Canada, 2019, pp. 86–91.

[25] Y. Bao, T. Xiong, Z. Hu, Multi-step-ahead time series prediction using multiple-output support vector regression, Neurocomputing 129 (2014) 482–493.

[26] A. Khan, X. Yan, S. Tao, N. Anerousis, Workload characterization and prediction in the cloud: a multiple time series approach, in: Proc. 2012 IEEE Network Operations and Management Symposium, Maui, HI, USA, 2012, pp. 1287–1294.

[27] F.J. Baldan, S. Ramirez-Gallego, C. Bergmeir, J.M. Benitez-Sanchez, F. Herrera, A forecasting methodology for workload forecasting in cloud systems, IEEE Transactions on Cloud Computing 6 (4) (2018) 929–941.

[28] S. Islam, J. Keung, K. Lee, A. Liu, Empirical prediction models for adaptive resource provisioning in the cloud, Future Generation Computer Systems 28 (1) (2012) 155–162.

[29] Y. Lu, J. Panneerselvam, L. Liu, Y. Wu, RVLBPNN: a workload forecasting model for smart cloud computing, Scientific Programming (2016) 1–9.

[30] R. Hu, J. Jiang, G. Liu, L. Wang, Efficient resources provisioning based on load forecasting in cloud, The Scientific World Journal (2014) 1–12.

[31] H. Fang, N. Tian, Y. Wang, M. Zhou, M.A. Haile, Nonlinear Bayesian estimation: from Kalman filtering to a broader horizon, IEEE/CAA Journal of Automatica Sinica 5 (2) (2018) 401–417.

[32] M. Ghahramani, Y. Qiao, M. Zhou, A.O. Hagan, J. Sweeney, AI-based modeling and data-driven evaluation for smart manufacturing processes, IEEE/CAA Journal of Automatica Sinica 7 (4) (Jul. 2020) 948–959.

[33] B. Song, Y. Yu, Y. Zhou, Z. Wang, S. Du, Host load prediction with long short-term memory in cloud computing, The Journal of Supercomputing 74 (12) (2018) 6554–6568.

[34] S. Barra, S.M. Carta, A. Corriga, A.S. Podda, D.R. Recupero, Deep learning and time series-to-image encoding for financial forecasting, IEEE/CAA Journal of Automatica Sinica 7 (3) (2020) 683–692.

[35] T.D. Pham, K. Wärdell, A. Eklund, G. Salerud, Classification of short time series in early parkinsons disease with deep learning of fuzzy recurrence plots, IEEE/CAA Journal of Automatica Sinica 6 (6) (2019) 1306–1317.

[36] P.M. Kebria, A. Khosravi, S.M. Salaken, S. Nahavandi, Deep imitation learning for autonomous vehicles based on convolutional neural networks, IEEE/CAA Journal of Automatica Sinica 7 (1) (2019) 82–95.

[37] G. Bao, Y. Zhang, Z. Zeng, Memory analysis for memristors and memristive recurrent neural networks, IEEE/CAA Journal of Automatica Sinica 7 (1) (2019) 96–105.

[38] Y. Ouyang, L. Dong, L. Xue, C. Sun, Adaptive control based on neural networks for an uncertain 2-dof helicopter system with input deadzone and output constraints, IEEE/CAA Journal of Automatica Sinica 6 (3) (2019) 807–815.

[39] H. Fei, F. Tan, Bidirectional grid long short-term memory (BiGridLSTM): A method to address context-sensitivity and vanishing gradient, Algorithms 11 (11) (2018) 172.

[40] Y. Lv, Y. Duan, W. Kang, Z. Li, F.Y. Wang, Traffic flow prediction with big data: a deep learning approach, IEEE Transactions on Intelligent Transportation Systems 16 (2) (2015) 865–873.

[41] G.M. Wang, J.F. Qiao, J. Bi, W.J. Li, M.C. Zhou, TL-GDBN: growing deep belief network with transfer learning, IEEE Transactions on Automation Science and Engineering 16 (2) (2019) 874–885.

[42] X. Shi, Z. Chen, H. Wang, D.Y. Yeung, W.K. Wong, W.C. Woo, Convolutional LSTM network: a machine learning approach for precipitation nowcasting, in: Proc. Advances in Neural Information Processing Systems, Montreal, Canada, 2015, pp. 802–810.

[43] J. Hanson, Y. Yang, K. Paliwal, Y. Zhou, Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks, Bioinformatics 33 (5) (2017) 685–692.

[44] M. Schuster, K.K. Paliwal, Bidirectional recurrent neural networks, IEEE Transactions on Signal Processing 45 (11) (1997) 2673–2681.

[45] D. Mozyrska, D.F. Torres, The natural logarithm on time scales, Journal of Dynamical Systems and Geometric Theories 7 (1) (2009) 41–48.

[46] S. Zhang, Z. Kang, Z. Hong, Z. Zhang, C. Wang, J. Li, Traffic flow prediction based on cascaded artificial neural network, in: Proc. IEEE International Geoscience and Remote Sensing Symposium, Valencia, Spain, 2018, pp. 7232–7235.

[47] C. Reiss, J. Wilkes, J.L. Hellerstein, Google cluster-usage traces: format+schema, Google Inc., 2011, pp. 1–14.

[48] C. Reiss, A. Tumanov, G.R. Ganger, R.H. Katz, M.A. Kozuch, Heterogeneity and dynamicity of clouds at scale: Google trace analysis, in: Proceedings of the Third ACM Symposium on Cloud Computing, 2012, pp. 1–13.

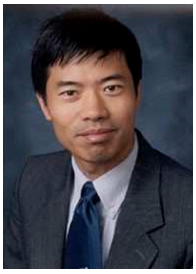
[49] M. Alam, K.A. Shakil, S. Sethi, Analysis and clustering of workload in google cluster trace based on resource usage, in: 2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on

Distributed Computing and Applications for Business Engineering (DCABES), 2016, pp. 740–747.
[50] G. Amvrosiadis, J.W. Park, G.R. Ganger, G.A. Gibson, E. Baseman, N. DeBardeleben, On the diversity of cluster workloads and its impact on research results, in: 2018 Annual Technical Conference, 2018, pp. 533–546.
[51] S. Gupta, A.D. Dileep, Long range dependence in cloud servers: a statistical analysis based on Google workload trace, Computing, 2020, pp. 1–19.
[52] S. Jachner, G. Van den Boogaart, T. Petzoldt, Statistical methods for the qualitative assessment of dynamic models with time delay (R package qualV), Journal of Statistical Software 22 (8) (2007) 1–30.
[53] S. Ohno, T. Shiraki, M.R. Tariq, M. Nagahara, Mean squared error analysis of quantizers with error feedback, IEEE Transactions on Signal Processing 65 (22) (2017) 5970–5981.
[54] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, J. Wang, Dendritic Neuron Model With Effective Learning Algorithms for Classification, Approximation, and Prediction, IEEE Transactions on Neural Networks and Learning Systems 30 (2) (2019) 601–614.
[55] A.C. Cameron, F.A.G. Windmeijer, An R-squared measure of goodness of fit for some common nonlinear regression models, Journal of Econometrics 77 (2) (1997) 329–342.
[56] M.H. Ghahramani, M. Zhou, C.T. Hon, Toward cloud computing QoS architecture: analysis of cloud systems and cloud services, IEEE/CAA Journal of Automatica Sinica 4 (1) (2017) 5–17.
[57] X. Luo, M.C. Zhou, Y. Xia, Q. Zhu, A.C. Ammari, A. Alabdulwahab, Generating highly accurate predictions for missing QoS data via aggregating nonnegative latent factor models, IEEE Transactions on Neural Networks and Learning Systems 27 (3) (2016) 524–537.
[58] X. Luo, M.C. Zhou, S. Li, M.S. Shang, An inherently nonnegative latent factor model for high-dimensional and sparse matrices from industrial applications, IEEE Transactions on Industrial Informatics 14 (5) (May 2018) 2011–2022.
[59] L. Huang, M. Zhou, K. Hao, Non-dominated immune-endocrine short feedback algorithm for multi-robot maritime patrolling, IEEE Transactions on Intelligent Transportation Systems 21 (1) (2020) 362–373.
[60] W. Dong, M. Zhou, A supervised learning and control method to improve particle swarm optimization algorithms, IEEE Transactions on Systems, Man and Cybernetics: Systems 47 (7) (2017) 1149–1159.

Jing Bi (M'13-SM'16) received her B.S., and Ph.D. degrees in Computer Science from Northeastern University, Shenyang, China. She was a Post-doc researcher at Department of Automation, Tsinghua University, Beijing, China. She was a research scientist at the Beijing Research Institute of Electronic Engineering Technology, Beijing, China. She was a research assistant and participated in research on cloud computing at the IBM Research, Beijing, China. She was a Visiting Research Scholar with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. She is currently an Associate Professor with the Faculty of Information Technology, School of Software Engineering, Beijing University of Technology, Beijing, China. She has over 100 publications in international journals and conference proceedings, including IEEE TNNLS, TCC, TASE, TSC, TII, TCYB, IoT Journal, and Elsevier INS, JPDC, Computers in Industry and Neurocomputing. Her research interests are in distributed computing, cloud computing, large-scale data analytics, machine learning and performance optimization. Dr. Bi was the recipient of the IBM Fellowship Award, the Best Paper Award in the 17th IEEE International Conference on Networking, Sensing and Control, and the First-Prize Progress Award of Chinese Institute of Simulation Science and Technology. She is now an Associate Editor of IEEE ACCESS. She is a senior member of the IEEE.

Shuang Li is currently a Master student in the Faculty of Information Technology, School of Software Engineering, Beijing University of Technology, Beijing, China. Before that, he received his B.S. degree in Software Engineering from Beijing University of Technology in 2018. His research interests include cloud computing, data center, energy management, big data, time series prediction, machine learning, and deep learning. Li was the recipient of the Best Paper Award-Finalist in the 16th IEEE International Conference on Networking, Sensing and Control.

Haitao Yuan (S'15-M'17) received the Ph.D. degree in Computer Engineering from New Jersey Institute of Technology (NJIT), Newark, NJ, USA in 2020. Before that, he received his Ph.D. degree in Modeling Simulation Theory and Technology from Beihang University, Beijing, China in 2016 and the M.S. and B.S. degrees in Software Engineering from Northeastern University, Shenyang, China, in 2010 and 2012, respectively. He is currently an Associate Professor with the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. He was an Associate Professor with the School of Software Engineering, Beijing Jiaotong University, Beijing, China. He was a Ph.D. student with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong from 2013 to 2014. He was also a visiting doctoral student with NJIT in 2015. He has over 80 publications in international journals and conference proceedings, including IEEE Internet of Things Journal, IEEE Transactions on Cloud Computing, IEEE Transactions on Automation Science and Engineering, IEEE Transactions on Services Computing, IEEE Transactions on Industrial Informatics and IEEE Transactions on Cybernetics. His research interests include cloud computing, edge computing, data centers, big data, machine learning, deep learning and optimization algorithms. Dr. Yuan was the recipient of the 2011 Google Excellence Scholarship, the recipient of the Best Paper Award-Finalist in the 16th IEEE International Conference on Networking, Sensing and Control (ICNSC), and the recipient of the Best Paper Award in the 17th ICNSC.



MengChu Zhou (S'88-M'90-SM'93-F'03) received his B. S. degree in Control Engineering from Nanjing University of Science and Technology, Nanjing, China in 1983, M.S. degree in Automatic Control from Beijing Institute of Technology, Beijing, China in 1986, and Ph. D. degree in Computer and Systems Engineering from Rensselaer Polytechnic Institute, Troy, NY in 1990. He joined New Jersey Institute of Technology (NJIT), Newark, NJ in 1990, and is now a Distinguished Professor of Electrical and Computer Engineering. He has over 900 publications including 12 books, 600+ journal papers (450+ in IEEE transactions), 27 patents and 29 book-chapters. His research interests are in Petri nets, intelligent automation, Internet of Things, big data, web services, and intelligent transportation. He is the founding Editor of IEEE Press Book Series on Systems Science and Engineering and Editor-in-Chief of IEEE/CAA Journal of Automatica Sinica. He is presently Associate Editor of IEEE Transactions on Intelligent Transportation Systems, IEEE Internet of Things Journal, and IEEE Transactions on Systems, Man, and Cybernetics: Systems. He is a recipient of Humboldt Research Award for US Senior Scientists from Alexander von Humboldt Foundation, Franklin V. Taylor Memorial Award and the Norbert Wiener Award from IEEE Systems, Man and Cybernetics Society, Excellence in Research Prize and Medal from New Jersey Institute of Technology, and Edison Patent Award from the Research & Development Council of New Jersey. He is a life member of Chinese Association for Science and Technology-USA and served as its President in 1999. He is a Fellow of International Federation of Automatic Control (IFAC), American Association for the Advancement of Science (AAAS) and Chinese Association of Automation (CAA).