The Journal of Supercomputing (2018) 74:6033–6068
https://doi.org/10.1007/s11227-018-2510-7

A joint feature selection framework for multivariate resource usage prediction in cloud servers using stability and prediction performance

Shaifu Gupta[1] · **A. D. Dileep**[1] · **Timothy A. Gonsalves**[1]

Published online: 27 July 2018

Abstract
Resource provisioning in cloud servers depends on future resource utilization of different jobs. As resource utilization trends vary dynamically, effective resource provisioning requires prediction of future resource utilization. The problem becomes more complicated as performance metrics related to one resource may depend on utilization of other resources also. In this paper, different multivariate frameworks are proposed for improving the future resource metric prediction in cloud. Different techniques for identifying the set of resource metrics relevant for the prediction of desired resource metric are analyzed. The proposed multivariate feature selection and prediction frameworks are validated for CPU utilization prediction in Google cluster trace. Joint analysis based on the prediction performance of the multivariate framework as well as its stability is used for selecting the most suitable feature selection framework. The results of the joint analysis indicate that features selected using the Granger causality technique perform best for multivariate resource usage prediction.

Keywords Load prediction · Cloud · Long-range dependence · Multivariate · Feature selection

1 Introduction

Cloud computing has revolutionized the computing capacities by providing on demand access to the virtual machines and hosted services over the Internet. It offers a networked, highly scalable and virtualized environment. A number of clients access the services hosted by a cloud dynamically at varying instants [22]. The service providers have to ensure optimal allocation of resources to their clients to satisfy their resource

✉ Shaifu Gupta
shaifugpt@gmail.com; shaifu_gupta@students.iitmandi.ac.in

[1] School of Computing and Electrical Engineering, Indian Institute of Technology Mandi, Kamand, Himachal Pradesh 175005, India

needs and achieve maximum profit [26]. However, the dynamic variations observed in the requests of resources make resource provisioning a challenging problem for the service providers.

Inefficiency in allocation of resources results in over- or under-provisioning issues [19]. Autoscaling is an important cloud computing paradigm that dynamically allocates resources to client applications to match their resource requirements. Hence, it avoids issues of idle resources and over allocation [10].

Predicting future resource utilization in advance aids the service administrators in better autoscaling by effectively managing and fulfilling the client requests. Future resource usage prediction may be modeled as a time series analysis problem. Different statistical methods have been proposed for predicting future resource usage in cloud environment. Gong et al. [12] and Nguyen et al. [30] use Markov chain models for prediction of future resource usage trends. In [24,41], autoregressive integrated moving average (ARIMA) models are used for resource utilization prediction in cloud. In [3], neural networks are used for prediction of future resource load. Most of these methods assume that the observations separated by a long time span are unrelated to each other. But the presence of dependence in distant observations has been observed in Ethernet traffic [23] and cloud workloads [16]. This phenomenon is called long-range dependence [28]. To model long-range dependence in cloud resource usage workloads, Gupta et al. [16] used fractional differencing extensions of different resource usage predictions methods and observed that fractional differencing-based extensions of resource usage prediction methods perform better for out-of-sample usage predictions. In [36], long short-term memory (LSTM) models are used for future resource usage predictions in cloud environment.

Many existing methods for resource usage prediction in cloud use univariate time series prediction models. These models use past trends in the desired resource metric only to generate future predictions. Univariate models may result in inaccurate predictions. For example, if the central processing unit (CPU) load is increasing linearly from 0 to 20% with time, we expect that it will continue to increase at the same rate to 40 and 60%. However, if we note that memory usage has increased from 10 to 90% during the same period, we may surmise that the system will soon run out of memory and start paging to disk. This will cause future CPU usage to decrease. This indicates that prediction of a resource metric depends not only on its past trends (as in univariate time series prediction), but also on the trends of other resource metrics as well. This brings forth the need of multivariate time series prediction.

Multivariate time series prediction analyzes the effect of other resource metrics on the predictions of the desired resource metric. In [17,25], only a limited set of metrics like CPU and memory usage are analyzed for future resource usage predictions in cloud. It is observed in [4] that studying many related variables together aids in better understanding by analyzing the variations in other metrics as well.

In this work, we propose multivariate frameworks for future resource usage prediction in cloud. A diverse set of features are analyzed. In this article, we use the terms features and resource metrics interchangeably. We use different features such as number of running jobs, memory usage, page cache, disk input/output (I/O) time, disk space and CPU usage for prediction of future CPU resource utilization. An important concern in multivariate prediction is the choice of a relevant set of features. We pro-

pose to analyze different feature selection techniques based on Pearson correlation, Spearman correlation, Granger causality and mutual information criterion to select the robust and relevant set of features for predicting future CPU utilization. To the best of our knowledge, the use of feature selection techniques for selecting the relevant set of features from a large and diverse set in the prediction of the desired resource metric has not been explored. Cloud resource usage workloads are highly dynamic in nature; therefore, the characteristics of resource usage patterns may vary in time. There is a need of a feature selection technique that is robust to the changes in the trends of resource usage data. In the present work, we propose to analyze the stability of different feature selection techniques. Stability quantifies the sensitivity of feature selection technique to the changes in the data. A combined analysis based on the prediction as well as its stability is used to select the robust feature selection technique. To the best of our knowledge, the use of a prediction and stability-based joint framework for selection of suitable feature selection techniques has not been explored for the prediction of resource usage trends in cloud. The proposed multivariate prediction frameworks are validated on the Google cluster trace [33].

The main contributions of this paper are: (i) design of multivariate time series prediction framework for PRESS, AGILE, ARIMA, NARNN, LSTM and BLSTM models for resource usage prediction, (ii) analysis and comparison of the performance of the proposed multivariate prediction models with state-of-the-art univariate prediction models, (iii) different methods for selection of relevant subset of features for future resource utilization prediction and (iv) a prediction and stability-based joint framework for selection of suitable feature selection techniques.

The remainder of the paper is organized as follows. In Sect. 2, we present an overview of the prominent approaches used for resource usage prediction in cloud. In Sect. 3, we present the proposed multivariate extensions of the prominent state-of-the-art methods. Section 4 explains different feature selection techniques to be used for the selection of relevant set of features. In Sect. 5, we present joint framework for selection of appropriate feature selection technique for prediction. Section 6 presents the dataset used in our studies for analyzing cloud workloads and prediction of future resource utilization. It also presents the results of different multivariate resource usage prediction models. Finally, Sect. 7 presents the conclusions and future work.

2 Related work

Different researchers have proposed different models for resource prediction in cloud network. In this section, we briefly describe some of the prominent models used for resource utilization prediction in cloud. Table 1 presents a survey of prominent models for resource usage prediction in cloud. It is seen from Table 1 that these methods can be broadly grouped into 4 categories such as (i) linear autoregressive models, (ii) nonlinear analysis methods, (iii) Markov models and (iv) fractional differencing-based methods. Fractional differencing-based methods are proposed to model the long- range dependence in cloud workloads. The models that come under this category are the fractional differencing-based extensions of methods in above three categories. Google cloud workloads have shown long- range dependence characteristics and methods

Table 1 Survey of works related to resource usage prediction models

Sr. No.	Broad categories	Techniques	References
1.	Linear autoregressive models	Autoregressive integrated moving average model (ARIMA),	[24,41]
		Exponential smoothing,	[20]
		AR, ARMA, ARIMA and exponential smoothing hybrid	[29]
2.	Nonlinear analysis methods	Feed-forward artificial neural network.	[2,3]
		Bayesian network,	[9,34]
		Support vector regression,	[19]
		Fuzzy neural network,	[5]
		Long short-term memory models (LSTM, BLSTM)	[15,36]
3.	Markov models	Predictive elastic resource scaling (PRESS),	[12]
		AGILE	[30]
4.	Fractional difference methods	Fractional ARIMA,	[16]
		Fractional neural network,	[16]
		Fractional PRESS,	[16]
		Fractional AGILE	[16]

that model long-range dependence have shown better prediction results [15,16,36]. In this section, we briefly describe some of the state-of-the-art models from each of the categories.

Gong et al. [12] proposed a method named predictive elastic resource scaling (PRESS). It uses state- and signature-driven approaches for generating future resource usage predictions. A discrete time Markov chain model is used for predicting future resource usage states. The load on the desired resource metric at different instants is divided into m discrete bins. The future usage at time $t+1$ is predicted by extracting the most probable future state of the resource load sequence. A transition probability matrix $\mathbf{P}$ of size $m \times m$ is created where $p_{i1,i2}$ denotes the conditional probability of transition from state $i1$ to $i2$. The probability of transition to a next state is computed using the Chapman–Kolmogorov equations as:

$$\boldsymbol{\pi}^t = \boldsymbol{\pi}^{t-1}\,\mathbf{P}. \tag{1}$$

Here $\boldsymbol{\pi}^t$ and $\boldsymbol{\pi}^{t-1}$ are the state probability vectors at time t and $t-1$, respectively. Here, $\boldsymbol{\pi}^t = [\pi_1^t, \pi_2^t, \ldots, \pi_m^t]$ where π_i^t is the probability of being in state i at time t. The load of the resource at time t is given by the state that has the highest probability at time t.

AGILE [30] extends PRESS by using wavelet-based method along with Markov models. The original time series is divided into different components called wavelets that carry meaningful signals of the original time series. Markov model (as implemented in PRESS) is used to generate future predictions for each decomposed signal. Then, the final out-of-sample predictions are obtained by reconstructing the predicted

signal from the predicted component wavelets. The predictions of AGILE are shown to be better than PRESS.

In [41], ARIMA method is used for extrapolating resource usage values. ARIMA is a linear method of prediction, where the resource usage at time t $\hat{x}^t$ is predicted as a weighted sum of its past lags. It is given as:

$$\hat{x}^t = a_0 x^{t-1} + a_1 x^{t-2} + \cdots + a_{p-1} x^{t-p} + b_0 \epsilon^{t-1} + \cdots + b_{q-1} \epsilon^{t-q}. \quad (2)$$

Here, x^t is the resource usage at time instant t. The terms ϵ are the error terms associated with the past lags in the model. The terms a, b denote the weights for the past lags and the error terms, respectively. The terms p and q are the number of previous lags in time and error terms, respectively. In [24], wavelets are used with the ARIMA model for CPU usage prediction in cloud.

Caglar and Gokhale [3] used a neural network model for generating future resource utilization predictions as:

$$\hat{x}^t = \sum_{j=1}^{q} \alpha_j g \left(\sum_{i=1}^{p} \beta_{ij} x^{t-i} \right). \quad (3)$$

Here, $\hat{x}^t$ is the predicted resource usage value at time instant t and p specifies the past lags used by the model. The term q denotes the number of neurons in the hidden layer. The function $g(.)$ is the activation function, and α, β are the weights of the neural network model. In contrast to an ARIMA model, neural networks have the ability to model and generalize both linear and nonlinear relationships between inputs and outputs [3].

Most of the above-mentioned works do not consider the presence of long-range dependence in the resource workloads. It is assumed that the time series is stationary and memory less. Gupta et al. [16] analyzed cloud workloads and observed the presence of long-range dependence in the resource usage time series. Fractional differencing method is applied to model long-range dependence in resource usage. The fractional differencing operator is defined as:

$$x^{t'} = x^t - d x^{t-1} + \frac{d(d-1)}{2!} x^{t-2} - \frac{d(d-1)(d-2)}{3!} x^{t-3} \ldots \quad (4)$$

where $x^{t'}$ denotes the value of resource usage at time t after fractional differencing and d is the difference parameter. In fractional differencing, the parameter d takes fractional values given by, $d = H - 0.5$ where H is the Hurst parameter. Fractional differencing-based extensions of PRESS, AGILE, ARIMA and neural network models are used for future CPU resource usage.

Song et al. [36] explored long short-term memory (LSTM) models for future resource usage prediction in cloud. LSTM models are a special type of recurrent neural networks. They can model long-range dependence in time series. In contrast to a traditional neural network, LSTM model provides connections between the units of hidden layers for capturing the sequential information. The network is divided into

blocks where each block has three gates. These gates regulate the flow of information into the LSTM memory. The three gates in an LSTM block are the input gate, the forget gate and the output gate. The LSTM block utilizes the present input as well as the past hidden layer information to determine the present output.

Most of the resource utilization prediction models use only the past trends in the resource usage of the desired resource metric for future resource prediction. However, in [25] a multi-resource prediction model using CPU and memory usage is analyzed where the future usage of a resource is predicted by analyzing the statistical relationship of the same resource at different time instants as well as the cross-correlation between different features.

$$\hat{x}^t = \sum_{i=1}^{p} a_i x^i + \sum_{i=1}^{p} b_i y^i. \tag{5}$$

The future usage of resource metric x, at time t, $\hat{x}^t$ is predicted from the history of other resource metric y as well as the history of x. Linear regression is used, and it is observed that analyzing the correlated features results in better predictions. In [17], CPU usage, memory usage and I/O are used for generating the future resource usage predictions using vector autoregressive models. It is shown in the context of sales forecast [42], prediction of mortality rates [32] and prediction of emergency department demand [4] that a multivariate model provides promising results over its univariate counterparts.

Correlation-based feature selection has been used in [7], where Pearson correlation is used for recognizing the set of relevant features for forecasting the demand of energy. Sun et al. [37] analyzed Granger causality method for multivariate time series analysis of water quality monitoring and Parkinsons telemonitoring. In [11,27] mutual information criterion is used for selection of a set of features for time series prediction in clinical analysis and electrical energy output prediction, respectively. To the best of our knowledge, the application of feature selection techniques has not been reported for multivariate resource usage prediction in cloud data center.

In the present work, the number of features under consideration is increased to include different important features like disk usage, disk I/O time, page cache activity, memory access per instruction and cycles per instruction as they can provide more information. Including a diverse set of features will aid in understanding the correlation between different features and provides a better insight when the time series values of a single feature depends on the other time series as well. In the proposed framework, different feature selection techniques are compared to generate better out-of-sample predictions.

In the next section, the proposed multivariate extensions of some of the state-of-the-art methods are presented.

3 Multivariate resource utilization prediction in cloud

Univariate resource utilization prediction models analyze the temporal patterns in the past history of the desired resource metric itself to predict future trends. However, better prediction capabilities in forecasting new values for the desired resource metric can

be achieved by considering the temporal variations in other related resource metrics in addition to the desired resource metric itself. Multivariate analysis involves analyzing more than one resource metrics together to analyze the relationships between them and their combined effect on the desired resource metric. They provide greater statistical power than analyzing individual features. Multivariate time series prediction approaches attempt to closely model the reality where each decision involves analysis of more than one variable [4]. These approaches use statistical methods to measure relationship among different features and correlate how each feature is important to generate final out-of-sample prediction of the desired resource metric. The terms resource metric and features are used interchangeably.

In this work, we analyze the effect of usage of different resources for future prediction of utilization of the desired resource metric. Let $\mathbf{X}$ is a multivariate time series describing the evolution of state of the system using different resources metrics $\mathbf{x}$ over time. Here, $\mathbf{X}$ is defined as:

$$\begin{aligned} &\mathbf{X} = (\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^t, \ldots, \mathbf{x}^T) \\ &\text{where } \mathbf{x}^t = [x_1^t, x_2^t, \ldots, x_D^t]^\top \ \forall t = 1, \ldots, T. \end{aligned} \tag{6}$$

T indicates the total number of observations and $\mathbf{x}_i^t$ is the observation of any resource metric i recorded at time instant t. Here, D denotes the number of resource metrics under study. Given the time series $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{t-1}$. In multivariate time series forecast, we predict the future value of resource x_j, as a combination of two functions $f_1(.)$ and $f_2(.)$ as,

$$\hat{x}_j^t = f_1(x_j^{t-1}) + \sum_{k=1, k\neq j}^{D} f_2(x_k^{t-1}). \tag{7}$$

Here $f_1(.)$ learns the relationship of the desired resource metric with itself, and $f_2(.)$ learns the relationship of the desired resource metric with the other related resource metrics.

In this work, we propose to extend the existing state-of-the-art univariate resource prediction models for generating multivariate time series predictions.

3.1 Resource utilization prediction using multivariate Markov models

We propose to extend the Markov model-based methods PRESS and AGILE proposed in [12,30] to their multivariate extensions called as PRESS-M and AGILE-M. In PRESS, resource usage of the desired resource metric is divided into m discrete bins. In PRESS-M, we consider multiple (here D) resource metrics. Each individual metric is divided into m bins. If T is the number of observations for every resource metric, then each bin is of size T/m and is defined as:

$$\mathbf{x}_i = (\mathbf{x}_{i1}, \mathbf{x}_{i2}, \ldots, \mathbf{x}_{im}) \ \ \forall i = 1, 2, \ldots, D. \tag{8}$$

Figure 1 presents the state flow diagram of univariate and multivariate Markov models. It is seen that a univariate Markov model analyzes the temporal transitions in the desired

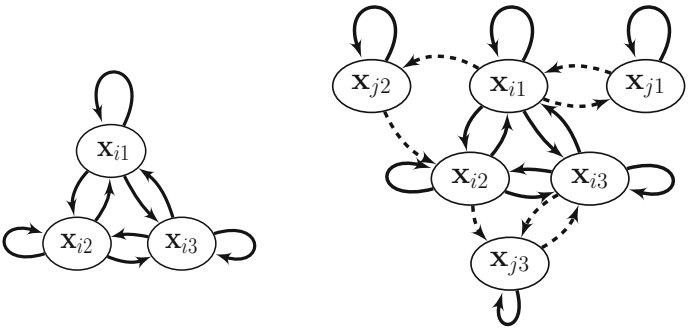


Fig. 1 Illustration of state flow diagram of **a** univariate **b** multivariate Markov model with 3 bins

resource metric, $\mathbf{x}_i$ alone. However, a multivariate Markov model helps to analyze the effect of other features as well (shown by dotted lines) on the desired resource metric. In contrast to PRESS, where transitions are considered from one bin to other within the desired resource metric itself, in PRESS-M the state probability distribution of the ith feature at time t depends on the state probabilities of other features (including itself) at time $t-1$. PRESS-M helps to analyze the effect of resource utilization of other features ($\mathbf{x}_j$) on the desired, to be predicted feature ($\mathbf{x}_i$). The value of the state transition probability from state i at time t is given as:

$$\boldsymbol{\pi}_i^t = \lambda_{ii}\mathbf{P}_{ii}\boldsymbol{\pi}_i^{t-1} + \sum_{j=1, j\neq i}^{D} \lambda_{ji}\mathbf{P}_{ji}\boldsymbol{\pi}_j^{t-1}.$$
$$\text{such that } \lambda_{ji} > 0 \text{ and } \sum_{j=1}^{D} \lambda_{ji} = 1 \text{ for } j = 1, \ldots, D \tag{9}$$

where D is the number of resource metrics under study and λ_{ji} are the weights for transitions from feature j to i. As given in Eq. 9, the state probability of the ith feature at time t depends on the weighted average of $\mathbf{P}_{ji}\boldsymbol{\pi}_j^{t-1}$ and $\mathbf{P}_{ii}\boldsymbol{\pi}_j^{t-1}$, where $\mathbf{P}_{ji}$ and $\mathbf{P}_{ii}$ denote the transition probability matrix of size $m \times m$ for transition from jth feature to ith feature and from ith feature to ith feature, respectively. The terms $\boldsymbol{\pi}_j^{t-1}$ and $\boldsymbol{\pi}_i^{t-1}$ are the state probabilities of the jth and ith features, respectively, to be in any state $(1, 2, \ldots, m)$ at time $t-1$ [6]. The resource usage of the ith feature is assigned to the state that has the maximum probability. In case of PRESS-M, we use the aforementioned multivariate Markov model for resource usage predictions.

For AGILE-M, wavelet transformation is combined with multivariate Markov model for future resource usage prediction. Wavelet transform provides the advantage that it captures both frequency and location information. In this method, wavelet decomposition is performed on time series of each resource metric and the decomposed wavelet components of each time series are passed to the multivariate Markov model for future prediction of each component. Figure 2 illustrates the wavelet decom-

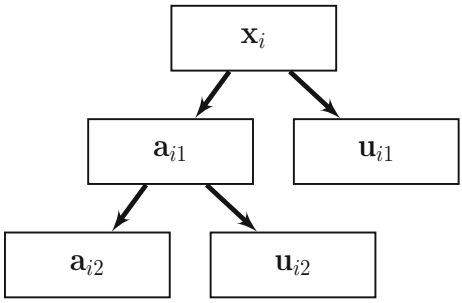


Fig. 2 Illustration of wavelet decomposition of resource metric $\mathbf{x}_i$ into coefficients of approximation, $\mathbf{a}_i$ and detail coefficients, $\mathbf{u}_i$ at scale = 2

position of a resource metric $\mathbf{x}_i$ into coefficients of approximation ($\mathbf{a}_i$) and detail ($\mathbf{u}_i$) at scale = 2. In the figure, $\mathbf{a}_{i1}$ and $\mathbf{a}_{i2}$ denote the approximation coefficients obtained after wavelet decomposition of resource metric $\mathbf{x}_i$ at levels 1 and 2, respectively. Similarly, $\mathbf{u}_{i1}$ and $\mathbf{u}_{i2}$ denote the detail coefficients obtained after wavelet decomposition of resource metric $\mathbf{x}_i$ at levels 1 and 2, respectively. As in AGILE [30], Daubechies family of wavelets is used for wavelet transform. After decomposition, the last level approximation and detail wavelet components of the individual features are concatenated to generate a D-dimensional matrix as:

$$\begin{aligned} \mathbf{A} &= [\mathbf{a}_{12}, \mathbf{a}_{22}, \ldots, \mathbf{a}_{i2}, \ldots, \mathbf{a}_{D2}] \\ \mathbf{D}_2 &= [\mathbf{u}_{12}, \mathbf{u}_{22}, \ldots, \mathbf{u}_{i2}, \ldots, \mathbf{u}_{D2}] \\ \mathbf{D}_1 &= [\mathbf{u}_{11}, \mathbf{u}_{21}, \ldots, \mathbf{u}_{i1}, \ldots, \mathbf{u}_{D1}]. \end{aligned} \tag{10}$$

Multivariate Markov model (as in PRESS-M) is used for generating out-of-sample predictions for each wavelet component. The final prediction for the desired resource metric is generated by reconstructing the original time series from the out-of-sample predictions of the individual components. Figure 3 presents an illustrative overview of the complete process where the original multivariate time series is decomposed into two detail signals and one approximation component at scale $= 2$ using Daubechies wavelet decomposition. For each component, the prediction is performed using multivariate Markov model and the final prediction is reconstructed using the predictions of individual components.

3.2 Resource usage prediction using multivariate ARIMA and neural network model

Vector autoregressive models (VAR) [35] are used as an extension of the univariate ARIMA models. In the present work, we name the multivariate ARIMA model used for future resource utilization prediction as ARIMA-M to maintain consistency with other multivariate extensions proposed in this paper. ARIMA-M model generalizes the ARIMA model by analyzing relations among more than one time series. It captures linear interdependencies in multivariate time series. The definition of ARIMA-M model is nearly identical to that of a ARIMA model as in Eq. 2 except that the resource usage x^t is replaced by a vector of different resource metrics, $\mathbf{x}^t$. It can be defined as:

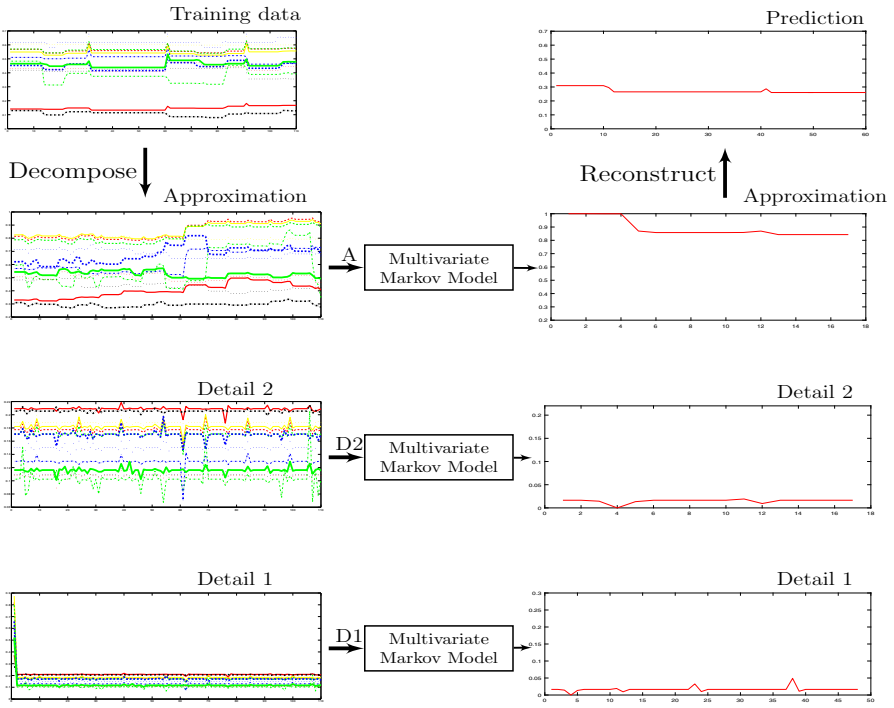


Fig. 3 Illustration of wavelet decomposition and reconstruction after prediction from a multivariate time series at scale = 2

$$\hat{\mathbf{x}}^t = \mathbf{A}_0\mathbf{x}^{t-1} + \mathbf{A}_1\mathbf{x}^{t-2} + \cdots + \mathbf{A}_{p-1}\mathbf{x}^{t-p} + \mathbf{B}_0\epsilon^{t-1} + \cdots + \mathbf{B}_{q-1}\epsilon^{t-q} \tag{11}$$

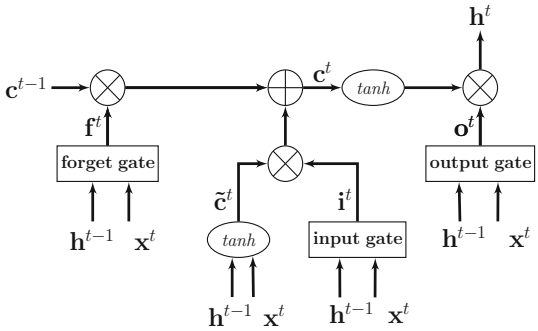
where $\mathbf{x}^1, \mathbf{x}^2 \ldots \mathbf{x}^{t-1}$ represent resource workload time series. The term $\hat{\mathbf{x}}^t$ is the predicted resource metric vector at time t. $\mathbf{A}$ and $\mathbf{B}$ are the $D \times D$ size weights given to the past lags in the time series, and ϵ are the error terms associated with the model. The terms p and q denote the number of past lags used in the model.

Similarly, multivariate nonlinear autoregressive neural network (NARNN-M) [38] is explored for generating the future resource usage predictions. The analytic equation of a multivariate neural network is similar to a univariate neural network except that each variable is replaced by a D-dimensional time series. It is given as:

$$\hat{\mathbf{x}}^t = \sum_{j=1}^{q} \boldsymbol{\alpha}_j g\left(\sum_{i=1}^{p} \boldsymbol{\beta}_{ij}\mathbf{x}_i\right) \tag{12}$$

where $\hat{\mathbf{x}}^t$ is the predicted vector at time instant t, p is the number of inputs, q is the number of neurons in the hidden layer, $g(.)$ is the activation function and $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ represent the connection weights that will be learned during training.

Fig. 4 Structure of LSTM-RNN



3.3 Resource usage predictions using multivariate LSTM (LSTM-M) and BLSTM (BLSTM-M) models

In this work, we propose to use multivariate LSTM for the prediction of resource utilization. LSTM models are a special type of neural networks that are well suited to learn and predict the time series when there is long-range dependence in the time series [18]. The LSTM unit consists of blocks where each block has three gates, namely input gate, output gate and the forget gate. Figure 4 shows the architecture of the LSTM block. The rectangular boxes in the figure represent a layer of sigmoidal neurons. The forget gate in the LSTM cell is formulated as:

$$\mathbf{f}^t = \text{sig}(\mathbf{W}_{xf}\mathbf{x}^t + \mathbf{W}_{hf}\mathbf{h}^{t-1} + \mathbf{b}_f) \tag{13}$$

where $\mathbf{f}^t$ is the output of the forget gate and sig(.) denotes the sigmoid activation function. $\mathbf{W}_{xf}$ and $\mathbf{W}_{hf}$ represent the weights between input ($\mathbf{x}^{t-1}$) and forget gate, weights between hidden layer ($\mathbf{h}^{t-1}$) and the forget gate, respectively. $\mathbf{b}_f$ represents the bias of the forget gate. In the next step, we need to decide what new information is to be stored in the cell state. This has two parts. First, the input gate decides which values are to be updated and next a vector of new candidate values, $\tilde{\mathbf{c}}^t$, is created to add new values to the state. The cell state $\mathbf{c}^t$ is computed as:

$$\mathbf{c}^t = \mathbf{f}^t\mathbf{c}^{t-1} + \mathbf{i}^t\tilde{\mathbf{c}}^t \tag{14}$$

where $\mathbf{c}^{t-1}$ represents the previous cell state at time $t-1$, $\mathbf{i}^t$ is the output of input gate and $\tilde{\mathbf{c}}^t$ is a vector of new candidate values given as:

$$\tilde{\mathbf{c}}^t = \tanh(\mathbf{W}_{xc}\mathbf{x}^t + \mathbf{W}_{hc}\mathbf{h}^{t-1} + \mathbf{b}_c) \tag{15}$$

where $\mathbf{W}_{xc}$ and $\mathbf{W}_{hc}$ represent the weights of cell state with the input and previous hidden layer, respectively. $\mathbf{b_c}$ represents the corresponding bias. The input gate of the cell $\mathbf{i}^t$ is defined as:

$$\mathbf{i}^t = \text{sig}(\mathbf{W}_{xi}\mathbf{x}^t + \mathbf{W}_{hi}\mathbf{h}^{t-1} + \mathbf{b}_i) \tag{16}$$

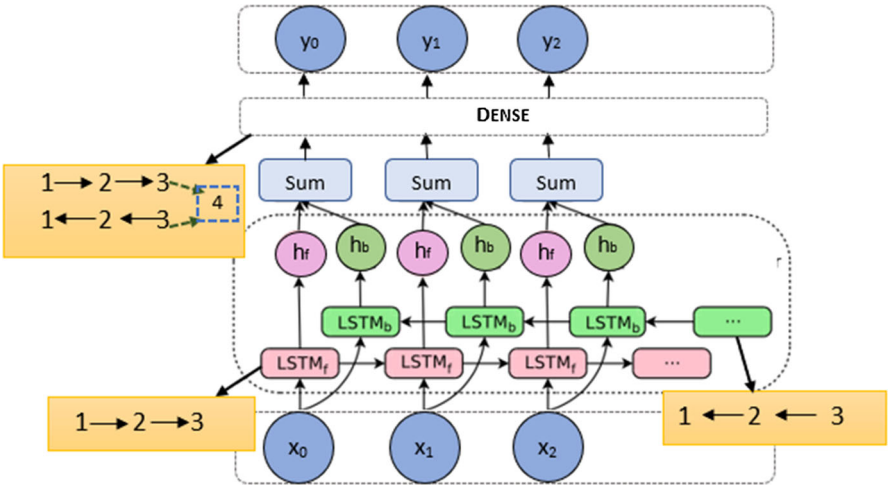


Fig. 5 Structure of bidirectional LSTM block. Illustration with sequence (1, 2, 3, 4)

where $\mathbf{W}_{xi}$ and $\mathbf{W}_{hi}$ are the weights associated with the inputs and the previous hidden layer for the input gate. $\mathbf{b}_i$ indicates the bias for the input gate. The output gate of the LSTM gate is defined as:

$$\mathbf{o}^t = \text{sig}(\mathbf{W}_{xo}\mathbf{x}^t + \mathbf{W}_{ho}\mathbf{h}^{t-1} + \mathbf{b}_o) \tag{17}$$

where $\mathbf{W}_{xo}$ represent weights for input in output gate and $\mathbf{W}_{ho}$ represent weights for past hidden layer in output gate. $\mathbf{b}_o$ represents the bias of the gate. The output of the hidden layer is computed as:

$$\mathbf{h}^t = \mathbf{o}^t \tanh(\mathbf{c}^t) \tag{18}$$

The output $\hat{\mathbf{x}}^t$ is computed as:

$$\hat{\mathbf{x}}^t = \mathbf{W}_{hy}\mathbf{h}^t + \mathbf{b}_y \tag{19}$$

where $\mathbf{W}_{hy}$ and $\mathbf{b}_y$ represent the weights and bias, respectively, between hidden layer and final output.

An LSTM accesses long-range context in window learning unidirectional flow (i.e., forward dependencies). We also propose to use bidirectional LSTM (BLSTM) that can access long-range context in window learning both forward and backward dependencies and hence enriches the understanding of resource workload patterns. Figure 5 shows the structure of BLSTM block. The data are processed in both directions where the forward layer iterates from $t = 1, \ldots, T$ and the backward layer iterates from $t = T, \ldots, 1$ learning the trends in the reverse time series. This bidirectional learning enhances the learning ability of the model. These models analyze the time series in both forward and reverse directions with two separate hidden layers ($\mathbf{h}_f$ and $\mathbf{h}_b$). For a sequence like (1, 2, 3, 4), the set of forward LSTM layers, LSTM_f, learns future temporal dependencies in the window ($1 \rightarrow 2 \rightarrow 3$) and the set of

backward LSTM layers, LSTM_b, learns past temporal dependencies in the window $(3 \rightarrow 2 \rightarrow 1)$. These hidden layers are then passed as input to the same output layer, DENSE, to predict the next value of the sequence. The output $\hat{\mathbf{x}}^t$ is computed as:

$$\hat{\mathbf{x}}^t = \mathbf{W}^t_{hy(bw)}\mathbf{h}^t_{bw} + \mathbf{W}^t_{hy(fw)}\mathbf{h}^t_{fw} + \mathbf{b}_y \tag{20}$$

where $\mathbf{h}^t_{bw}$ and $\mathbf{h}^t_{fw}$ represent the output of the backward and forward hidden layers at time t. $\mathbf{W}_{hy(bw)}$ and $\mathbf{W}_{hy(fw)}$ denote the weights between the final output and the backward and forward hidden layers, respectively. $\mathbf{b}_y$ represents the bias.

The multivariate resource load prediction models are combined with different feature selection techniques to identify the relevant set of features for resource workload prediction. The next section presents different feature selection techniques for selection of a relevant subset of features (resource metrics) from the universal set.

4 Feature selection framework for resource usage prediction

A complex system such as a cloud has 10s to 100s of performance metrics or features. Many are relatively independent of one another. A few are correlated or have a cause–effect relationship.

For an efficient and accurate multivariate prediction, it is desirable to identify the set of highly related features. By using all these, the model is more accurate, and by using only these, the time and space complexity are kept to computable levels.

A domain expert could select this set of features after careful consideration of a given system. However, such experts are hard to find and the effort needed for many clouds under diverse workloads is infeasible. Hence, our goal is to design a feature selection framework that uses several automatic algorithms to select the most relevant set of features.

In feature selection, a set of the most relevant l features are chosen from a set of D features such that $l \ll D$. Exhaustive search among the 2^D candidate subsets using a predictive model is computationally intensive and is infeasible for large D [8].

Some features have a causal relation. A change in one feature, the cause, precedes and causes a change in other feature, the effect. Other features are correlated but do not have a causal relation. Correlation does not imply that one variable causes other. For example, when there is congestion in a network, both the packet loss and the round trip average (RTA) values are high. But this does not imply that packet loss causes RTA to be high or vice versa. Both of them are manifestations of high load on the network. In our feature selection framework, we include four techniques. Two of these, Pearson correlation and Spearman correlation, select the relevant features based on correlation of the candidate resource metric with the desired resource metric. The other methods, Granger causality, select features based on causality and mutual information selects based on information shared between features. All of these techniques compute pairwise relations between the candidate resource metric with the desired resource metric which is repeated for all candidate resource metrics.

4.1 Pearson correlation-based feature selection

Pearson's correlation measures the linear relationship between two features where a change in one feature is associated with a proportional change in the other feature. It is the covariance of two features divided by the product of their standard deviations [7]. The Pearson correlation coefficient, r_{pear}, between two resource metrics $\mathbf{x}_i$ and $\mathbf{x}_j$ is given by

$$r_{\text{pear}} = \text{corr}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{t=1}^{T}(x_i^t - \mu_i)(x_j^t - \mu_j)}{\sqrt{\sum_{t=1}^{T}(x_i^t - \mu_i)^2}\sqrt{\sum_{t=1}^{T}(x_j^t - \mu_j)^2}} \tag{21}$$

where μ_i and μ_j represent the average value of the features (resource metrics) $\mathbf{x}_i$ and $\mathbf{x}_j$, respectively. Here any resource metric $\mathbf{x}_i$ is given as, $\mathbf{x}_i = [x_i^1, x_i^2, \ldots, x_i^t, \ldots, x_i^T]$. Pearson correlation gives a value in $[-1, +1]$, where $+1$ denotes maximum positive correlation and -1 specifies negative correlation between the two variables. If the two variables are independent of each other, then their correlation value is zero. The Pearson correlation coefficient is symmetric: $\text{corr}(\mathbf{x}_i, \mathbf{x}_j) = \text{corr}(\mathbf{x}_j, \mathbf{x}_i)$.

To select significant features from a set of features, a subset Z is redefined recursively. Initially, $Z = \{\}$. The candidate resource metric whose Pearson correlation coefficient value for the desired resource metric is greater than a threshold is selected as:

$$\begin{aligned} &Z = Z \cup \{\mathbf{x}_j\}, \;\; Z \in \mathbb{R}^{T \times l} \\ &\quad \text{where} \;\; \text{corr}(\mathbf{x}_i, \mathbf{x}_j) > \text{threshold} \; \forall j = 1, 2, \ldots, D \end{aligned} \tag{22}$$

4.2 Spearman correlation-based feature selection

Spearman correlation measures the strength and direction of monotonic association between two variables which tend to change together but not necessarily at a constant rate. Let $\mathbf{x}_i$ and $\mathbf{x}_j$ be the two resource metrics having T observations each. A rank of each value in the resource metrics $\mathbf{x}_i$ and $\mathbf{x}_j$ is obtained by assigning a rank of 1 to the lowest value, 2 to the next lowest and so on. The Spearman correlation coefficient is computed as:

$$r_{\text{spear}} = 1 - \frac{6\sum_{t=1}^{T}(c^t)^2}{T(T^2 - 1)} \tag{23}$$

where c^t is the difference in ranks of two features [40] at time t. The sign of the Spearman correlation indicates the direction of association between $\mathbf{x}_i$ and $\mathbf{x}_j$. Spearman correlation gives a value between $+1$ and -1 (both inclusive). If time series $\mathbf{x}_j$ tends to increase when time series $\mathbf{x}_i$ increases, the Spearman correlation coefficient is positive. If $\mathbf{x}_j$ tends to decrease when $\mathbf{x}_i$ increases, the Spearman correlation coefficient is negative. A Spearman correlation of zero indicates that there is no tendency for $\mathbf{x}_j$ to either increase or decrease when $\mathbf{x}_i$ increases. When $\mathbf{x}_i$ and $\mathbf{x}_j$ are perfectly monotonically related, the Spearman correlation coefficient becomes 1. Similar to Pearson

correlation coefficient, we select a subset of features Z whose correlation with the desired resource metric is greater than threshold.

4.3 Granger causality based feature selection

According to Granger causality, a time series $\mathbf{x}_j$ is said to Granger-cause a time series $\mathbf{x}_i$ ($\mathbf{x}_j \rightarrow \mathbf{x}_i$), if the predictions of $\mathbf{x}_i$ based on its own past values and on the past values of $\mathbf{x}_j$ are better than predictions of $\mathbf{x}_i$ based only on its own past values [37]. Causal relations are asymmetrical in nature, which implies that if $\mathbf{x}_j$ Granger-cause $\mathbf{x}_i$, then it is not necessary that $\mathbf{x}_i$ also Granger-cause $\mathbf{x}_j$. Granger causality is computed by formulating two linear regressions: one that predicts $\hat{x}_i^t$ using the observations from $\mathbf{x}_i$ alone and another that predicts $\hat{x}_i^t$ using observations from both $\mathbf{x}_i$ and $\mathbf{x}_j$. We refer them as $\hat{x}_{i(\text{res})}^t$ and $\hat{x}_{i(\text{unres})}^t$, respectively. These are given as:

$$\begin{aligned}\hat{x}_{i(\text{res})}^t &= a_0 x_i^{t-1} + a_1 x_i^{t-2} + \cdots + a_{p-1} x_i^{t-p} \\ \hat{x}_{i(\text{unres})}^t &= a_0 x_i^{t-1} + a_1 x_i^{t-2} + \cdots + a_{p-1} x_i^{t-p} + b_0 x_j^{t-1} + b_1 x_j^{t-2} + \cdots + b_{p-1} x_j^{t-p}.\end{aligned} \tag{24}$$

Here, p is the number of past lags. The terms a and b are the weights given to past lags of $\mathbf{x}_i$ and $\mathbf{x}_j$, respectively. F-test is applied to test the overall significance of linear regressions. The F-statistic is computed using the following equation:

$$\begin{aligned} F\text{-statistic} &= \frac{\text{SSE}_{\text{res}} - \text{SSE}_{\text{unres}}}{\text{SSE}_{\text{unres}}} \left(\frac{T - (2p+1)}{p} \right) \\ \text{where } \text{SSE}_{\text{res}} &= \sum_{t=1}^{T} (\hat{x}_{i(\text{res})}^t - x_i^t)^2 \text{ and } \text{SSE}_{\text{unres}} = \sum_{t=1}^{T} (\hat{x}_{i(\text{unres})}^t - x_i^t)^2 \end{aligned} \tag{25}$$

The terms SSE_{res} and $\text{SSE}_{\text{unres}}$ are the sum of square error in the prediction of $\hat{x}_i^t$ using the restricted model (using $\mathbf{x}_i$ alone) and the unrestricted model (using $\mathbf{x}_i$ and $\mathbf{x}_j$). Here, x_i^t denotes the actual workload of resource metric i at time t. All the features that have F-statistic value greater than a threshold value are chosen in the set of most relevant features.

4.4 Mutual information

Mutual information (MI) quantifies the amount of information shared between features. It measures how much knowledge of one feature decreases uncertainty about the other [11]. Mutual information measures nonlinear relationships between two features, $\mathbf{x}_i$ and $\mathbf{x}_j$. It is given as:

$$\text{MI}(\mathbf{x}_i, \mathbf{x}_j) = H(\mathbf{x}_i) + H(\mathbf{x}_j) - H(\mathbf{x}_i, \mathbf{x}_j). \tag{26}$$

Here, $H(\mathbf{x}_i)$ represents the entropy for the feature $\mathbf{x}_i$ and it measures uncertainty in $\mathbf{x}_i$. It is given as:

$$H(\mathbf{x}_i) = \sum_{t=1}^{T} p(x_i^t)\log(p(x_i^t)) \tag{27}$$

where $p(x_i^t)$ is the probability distribution of the feature x_i^t. The joint entropy of two features $H(\mathbf{x}_i, \mathbf{x}_j)$ is given as:

$$H(\mathbf{x}_i, \mathbf{x}_j) = \sum_{t=1}^{T}\sum_{t=1}^{T} p(x_i^t, x_j^t)\log(p(x_i^t, x_j^t)) \tag{28}$$

where $p(x_i^t, x_j^t)$ is the joint probability distribution of the features x_i^t and x_j^t. Mutual information between two features lies in the range $[0-\infty]$. If $\mathbf{x}_i$ and $\mathbf{x}_j$ are dependent, they will have large MI. This means that $\mathbf{x}_j$ contains some information about $\mathbf{x}_i$, indicating that it is a relevant feature. Mutual information ranks the features based on the amount of information shared with the desired resource metric. We select the features whose MI with the desired resource metric $\mathbf{x}_i$ is greater than threshold, $threshold$. If $\text{MI}(\mathbf{x}_j, \mathbf{x}_i) >$ threshold, then the feature $\mathbf{x}_j$ is selected as a relevant feature, otherwise it is removed.

We analyze different multivariate frameworks based on the aforementioned feature selection techniques to study their performance for resource utilization prediction. In addition to prediction performance, stability of a feature selection technique is also analyzed. The next section presents the method to analyze the stability of feature selection technique and the need of joint framework for selection of feature selection techniques.

5 Joint framework for selection of feature selection techniques

Different feature selection techniques model different trends from the data. Each technique may select a different subset and a different number of features. Therefore, selection of an appropriate feature selection technique is very important.

It is desirable to choose a feature selection technique that not only models well the important trends in the data but also suits the environment in which it is applied. Cloud environment is a highly dynamic environment where different numbers of users place varying service requests at different times. To model the dynamism in workloads of cloud applications, we identify two desired characteristics for the feature selection techniques. These are (i) prediction performance and (ii) stability.

The performance of out-of-sample predictions enables us to evaluate the effectiveness of a feature selection technique for modeling the trends in the data. It provides a measure to compare different feature selection techniques and select a subset of features that give more information about the desired resource metric than any other combination.

In addition to comparing the performance of out-of-sample predictions, stability of a feature selection technique is analyzed. Stability of a feature selection technique

defines its sensitivity to variations in the data. It is possible that selected features are best for one subset of data and may not be the best for other subset of data. Since cloud workloads are highly time varying, the characteristics of data may frequently change with time. Identifying run-time variations in data and controlling the selection of relevant features is difficult. Therefore, in this work we investigate the sensitivity of feature selection techniques on different subsets of data using Jaccard similarity index [31]. Jaccard similarity score is used to quantify the sensitivity of feature selection technique to changes in data. It is measured as the ability of a feature selection technique for selecting same features in different subsets of data. The Jaccard similarity index ϕ, between two subsets of data 1 and 2, is computed as:

$$\phi_{1,2} = \frac{c_{1,2}}{k_1 + k_2 - c_{1,2}} \tag{29}$$

Here, $c_{1,2}$ is the number of features selected in subset of data 1 and data 2. The terms, k_1 and k_2, denote the cardinality of the selected feature subsets from data 1 and data 2. The higher the similarity score, the more robust is the feature selection to the variations in the data.

To determine the stability of feature selection technique, the resource usage time series is divided into different overlapping blocks. The number of samples in each block is kept same. Fixed overlap partitions algorithm [39] is used to generate overlapping blocks having the same number of samples and a specified degree of overlap between the blocks. It is studied in [1] that without controlling the degree of overlap, it is difficult to be sure that the two selected feature subsets are different due to changes in the underlying data or the stability issues of the feature selection technique. Therefore, the stability of the feature selection technique is analyzed at different overlap ratios of 0, 0.25, 0.5 and 0.75.

Let $\mathbf{X} = (\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^t, \ldots, \mathbf{x}^T)$ be the resource usage time series, where T is the number of samples. Each $\mathbf{x}^t \in \mathbb{R}^D, \quad \forall t = 1, \ldots, T$. The series is divided into different overlapping blocks, i.e., $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_e, .$ Here, $\mathbf{X}_1 = \{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^t\}$, $\mathbf{X}_2 = \{\mathbf{x}^{t-1}, \ldots, \mathbf{x}^{2t-2}\}, \ldots, \mathbf{X}_e = \{\mathbf{x}^{(e-1)t-(2e-3)}, \ldots, \mathbf{x}^T\}$ with an overlap of 2. The feature selection techniques are applied to these blocks to output the feature subset for each of the block.

Let $\mathbf{S}$ be the set of features selected by any feature selection technique. $\mathbf{S}$ is a binary matrix of size $[e \times D]$. It has ones corresponding to features selected by the technique and zeros for remaining. The binary matrix $\mathbf{S}$ is given as:

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1^\top \\ \mathbf{s}_2^\top \\ \vdots \\ \mathbf{s}_e^\top \end{bmatrix} \tag{30}$$

where $\mathbf{s}_j$ is the subset of features selected by a feature selection technique for block j from training set, train. Jaccard similarity measure is used to compute the similarity, i.e., the consistency score between the features selected in different blocks with the

train set. The Jaccard similarity is computed between the train set and all the blocks $j = 1, \ldots, e$ using Eq. 29. The final consistency score (ϕ) of a feature selection technique is obtained by computing the average similarity score over all the blocks as:

$$\phi = \frac{\phi_{\text{train},1} + \phi_{\text{train},2} + \cdots + \phi_{\text{train},e}}{e} \tag{31}$$

For an illustration, let $\mathbf{X}_1$, $\mathbf{X}_2$ and $\mathbf{X}_3$ be the overlapping 3 blocks from the training set. Assume D be 4. Let $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$ be the vectors with one for selected features and zero for the remaining features corresponding to $\mathbf{X}_1$, $\mathbf{X}_2$ and $\mathbf{X}_3$. Then, the binary matrix $\mathbf{S}$ is given as:

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1^\top \\ \mathbf{s}_2^\top \\ \mathbf{s}_3^\top \end{bmatrix} = \begin{bmatrix} 1\,0\,0\,1 \\ 1\,1\,1\,1 \\ 1\,1\,0\,1 \end{bmatrix} \tag{32}$$

Here, the Jaccard similarity $\phi_{1,2}$ is given as:

$$\phi_{1,2} = \frac{2}{2+4-2} = 0.50 \tag{33}$$

Similarly, $\phi_{2,3}$ and $\phi_{1,3}$ are computed as:

$$\phi_{2,3} = \frac{3}{3+4-3} = 0.75 \quad \text{and} \quad \phi_{1,3} = \frac{2}{2+3-2} = 0.67 \tag{34}$$

The final consistency score ϕ is given by:

$$\phi = \frac{0.50 + 0.75 + 0.67}{3} = 0.41 \tag{35}$$

Both properties, prediction performance and stability, are highly important for modeling dynamism of resource utilization workloads in cloud environment. Hence, their joint analysis-based feature selection framework is proposed for selection of most suitable features for multivariate resource utilization prediction. The experiments and studies to analyze the performance of different frameworks are validated on Google cluster trace and are presented in next section.

6 Experimental studies

To evaluate the effectiveness of our multivariate prediction framework, we apply it to the Google cluster trace [33]. As the CPU is a key resource, we focus on prediction of CPU usage. The experimental methodology and results are described below.

Table 2 Resource metrics used for future CPU usage prediction

Feature No.	Resource metrics	Description
1	JOBS	Total number of jobs running in the cluster
2	CPU	Aggregate CPU usage of all jobs in the cluster
3	MEM	Aggregate memory usage of all jobs in the cluster
4	VM	Assigned memory usage but not necessarily used
5	UPC	Aggregate unmapped page cache in the cluster
6	TPC	Aggregate total page cache usage in the cluster
7	MAXM	Maximum memory usage observed over interval
8	DIO	Disk input/output time sum across all disks
9	DSP	Disk capacity space usage across all nodes
10	MAXC	Maximum CPU usage observed over interval
11	MAXD	Maximum disk I/O time observed over interval
12	CPI	Cycles per instruction across all nodes
13	MAI	Memory accesses per instruction across all nodes

6.1 Dataset

The Google cluster trace [33] presents resource workload data of a cluster of 12,500 machines during May 2011. It provides run-time information of different jobs arriving to the cluster for 29 days. The workload in this trace provides data on the arrival, execution and completion of different jobs. The arriving jobs are scheduled on different heterogeneous machines of the cluster. There are a variety of job types and heterogeneous machine types. Every job in the dataset is accompanied by a set of resource request and resource usage metrics at different time instants.

In this work, the resource usage metrics are analyzed to predict the expected resource workload on the cluster. CPU usage is one of the key resource metrics and is usually the first place we look when we observe the sign of jobs slowing down. Therefore, we focus on prediction of future CPU usage trends. In addition to CPU usage, several other resource metrics are used for multivariate CPU usage prediction. The different resource metrics used in this study are presented in Table 2. These resource metrics are chosen from the resource usage table of Google cluster trace. This table in the trace provides run-time resource usage information of different jobs running at different time instants. In the present work, all the 13 monitored resource usage metrics are chosen for analyzing their behavior during the study. Further, feature selection is performed to select the most relevant features for prediction.

In the present study, we divide the data in training, validation and test set. The fine-tuning of parameters for different models is carried using validation set, and the best suitable parameters for each of the respective models are used for generating future predictions. The 7 days of resource workload data is used for training and validating the performance of load prediction models. Resource usage values are aggregated at 10 second time interval. The out-of-sample future predictions are generated for the

next 10, 20 and 30 minutes. We use a machine with 40 Intel Xeon CPU cores and 128 GB RAM for the training of prediction models.

6.2 Metric for validation of actual resource workload predictions

Root-mean-square error (RMSE) is used to evaluate the accuracy of prediction results defined as follows:

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{P}(x_j^t - \hat{x}_j^t)^2}{P}} \tag{36}$$

where P is the prediction length and j is the desired resource metric. Here, we evaluate prediction models at $P = 60$, 120 and 180 steps. x_j^t denotes the actual resource load value at time instant t, and $\hat{x}_j^t$ denotes the resource load predicted at time instant t.

6.3 Long-range dependence

Long-range dependence is a phenomenon where an increase/decrease in the next step value is affected by several past time lags in the series [28]. Standard time series prediction models like vector autoregressive models cannot capture long-term (low frequency) dependence. They can only describe the short term (high- frequency behavior) of a time series. A stationary process is said to be long range dependent if

$$\lim_{k \longrightarrow \infty} \frac{\rho(k)}{c_p k^{-\alpha}} = 1 \tag{37}$$

where k is lag, $\rho(k)$ denotes the autocorrelation of resource load, α is a real number $\in (0, 1)$ and c_p is a constant such that $c_p > 0$. This implies that $\rho(k) \propto \frac{1}{k^\alpha}$, and hence, the dependence between consecutive observations exhibits power law decay as compared to a normal time series [14].

We analyzed the presence of long-range dependence in the resource workload by examining autocorrelation in the time series patterns. The autocorrelation of CPU usage is shown in Fig. 6a. We analyzed the exponential and power law nature of CPU autocorrelation. The general exponential equation used is:

$$f(k) = u * \exp(v * k). \tag{38}$$

The general power law equation used is:

$$f(k) = u * k^v. \tag{39}$$

We observed that the exponential distribution matches the autocorrelation trend at $u = 0.5571$ and $v = -0.001138$ with a goodness of fit of 0.08 in the root-mean-square error. The power law distribution closely matches the autocorrelation trend at $u = 1.848$ and $v = -0.3099$ with a goodness of fit of 0.0485 in the root-mean-square error (RMSE). This shows that the autocorrelation follows power law decay rather

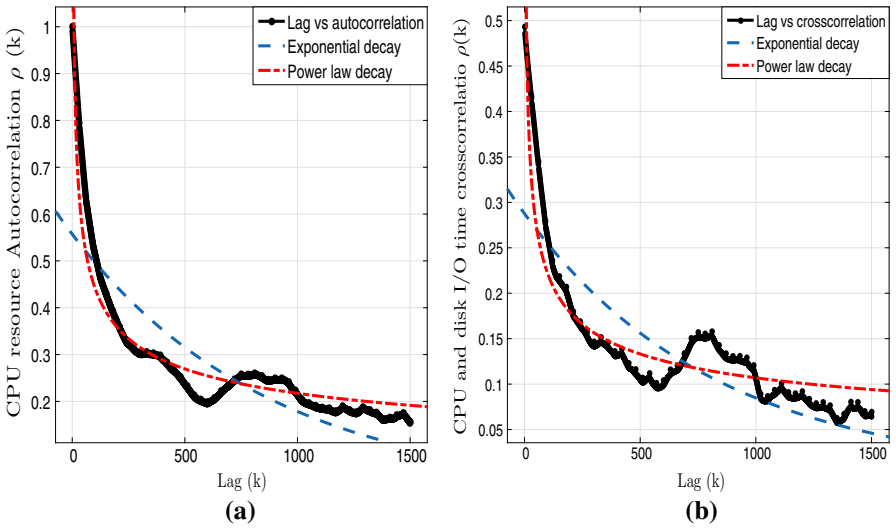


Fig. 6 **a** Autocorrelation plot of CPU usage. **b** Cross-correlation plot showing cross-correlation between CPU and disk I/O time

than exponential decay, hence showing the presence of long-range dependence in the CPU usage. Similar to autocorrelation, the cross-correlation behavior of CPU with other resource metrics is analyzed. Figure 6b shows the cross-correlation between the CPU usage, and the disk I/O time also exhibits power law decay. We observe that the exponential distribution matches the autocorrelation trend at $u = 2240$ and $v = -1.633e - 05$. The goodness of fit of exponential distribution is 12.79. The power law distribution closely matches the cross-correlation trend at $u = 2335$ and $v = -0.008538$. The goodness of fit for the power law distribution has a RMSE of 7.329. As there exists a slow decay in different consecutive lags, it indicates the presence of long-range dependence in resource load values. A similar behavior is noted for cross-correlation between CPU usage and other resource metrics as well.

The strength of long-range dependence in the resource load time series is determined using the rescaled range analysis method [21]. Rescaled range analysis method quantifies the relative persistence of long range in a time series. In this method, the resource load is divided into blocks of different sizes. The slope of the rescaled range analysis plot gives Hurst exponent (H) [13]. We observed the Hurst exponent of the CPU usage as 0.9029. As the Hurst exponent is greater than 0.5, it indicates the presence of long-range dependence in the data. In addition to CPU resource usage metrics, we computed the Hurst exponent value for other resource and performance metrics. We observed that different features exhibit different indices of long-range dependence $H = \{H1, H2, H3, \ldots\}$. To model the long-range dependence, each feature is differenced with its corresponding difference parameter, d_i where $d_i = H_i - 0.5$ for the ith feature.

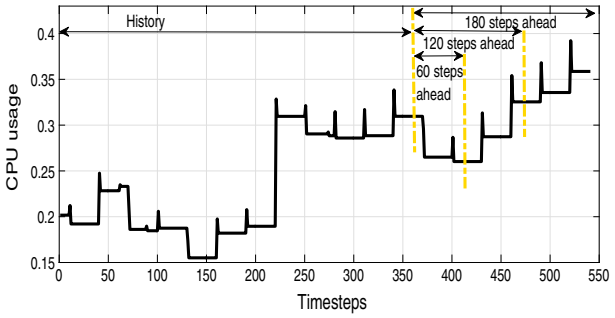


Fig. 7 Actual CPU usage with 360 steps of history and 60, 120 and 180-step-ahead trend

6.4 CPU usage prediction using all features

We applied our multivariate framework based on PRESS-M, AGILE-M, ARIMA-M and NARNN-M with fractional differencing for predicting future CPU usage to model long-range dependence in these workloads. We term them as *f*PRESS-M, *f*AGILE-M, *f*ARIMA-M and *f*NARNN-M. We also use multivariate LSTM and BLSTM models to generate the out of sample predictions for future CPU usage. Figure 7 shows the trend of CPU usage. The plot shows the 360 steps of history and 60 step (10 minutes), 120 step (20 minutes) and 180-step-ahead (30 minutes) CPU usage trends. The effect of all features (presented in Table 2) is analyzed for predicting out-of-sample CPU predictions. Table 3 presents the root-mean-square error (RMSE) of the multi-step-ahead predictions generated using univariate and multivariate models. It is observed that the error in the predictions generated by multivariate models using all features is less than that of error generated by univariate prediction models. Multivariate frameworks analyze correlations of the desired resource metric with other features and hence capture the dynamics of the underlying time series well. It is also seen that the BLSTM-based method outperforms the other resource utilization prediction methods. Figure 8 presents the predictions of *f*PRESS, *f*AGILE, *f*ARIMA, *f*NARNN, LSTM and BLSTM models for univariate and multivariate time series predictions using all 13 features (resource metrics). The predictions of univariate and multivariate models are compared at 60, 120 and 180 steps ahead. It is seen that the multivariate resource prediction models generate better out-of-sample predictions at different steps than their univariate counterparts.

We also experiment with a weighted resource usage prediction method. The prediction of the desired resource metric is obtained by using a weighted combination of other features. Bivariate models of CPU load are built with every other feature to assess the effect of other features on the predictions of CPU usage. The weights for the individual features are decided based on their performance on validation set. The feature that generates better prediction of the desired metric gets more weight as compared to others.

Figure 9 shows the weights given by different models to different features. It is observed that some features, namely maximum memory usage observed over a given interval (MAXM), disk space used (DSP) and memory usage (MEM), have been

Table 3 RMSE of CPU usage prediction in univariate and multivariate prediction in Google cluster trace

	Univariate			All features			Weighted individual features		
	60 steps	120 steps	180 steps	60 steps	120 steps	180 steps	60 steps	120 steps	180 steps
*f*PRESS	0.0260	0.0558	0.0753	0.0214	0.0548	0.0731	0.0187	0.0546	0.0708
*f*AGILE	0.0159	0.0535	0.0732	0.0139	0.0361	0.0686	0.0132	0.0361	0.0687
*f*ARIMA	0.0198	0.0308	0.0562	0.0158	0.0299	0.0555	0.0143	0.0285	0.0549
*f*NARNN	0.0135	0.0301	0.0551	0.0130	0.0289	0.0535	0.0126	0.0283	0.0534
LSTM	0.0123	0.0258	0.0434	0.0105	0.0201	0.0362	0.0103	0.0198	0.0359
BLSTM	**0.0115**	**0.0210**	**0.0385**	**0.0095**	**0.0184**	**0.0255**	**0.0092**	**0.0182**	**0.0250**

Bold indicates the best performance

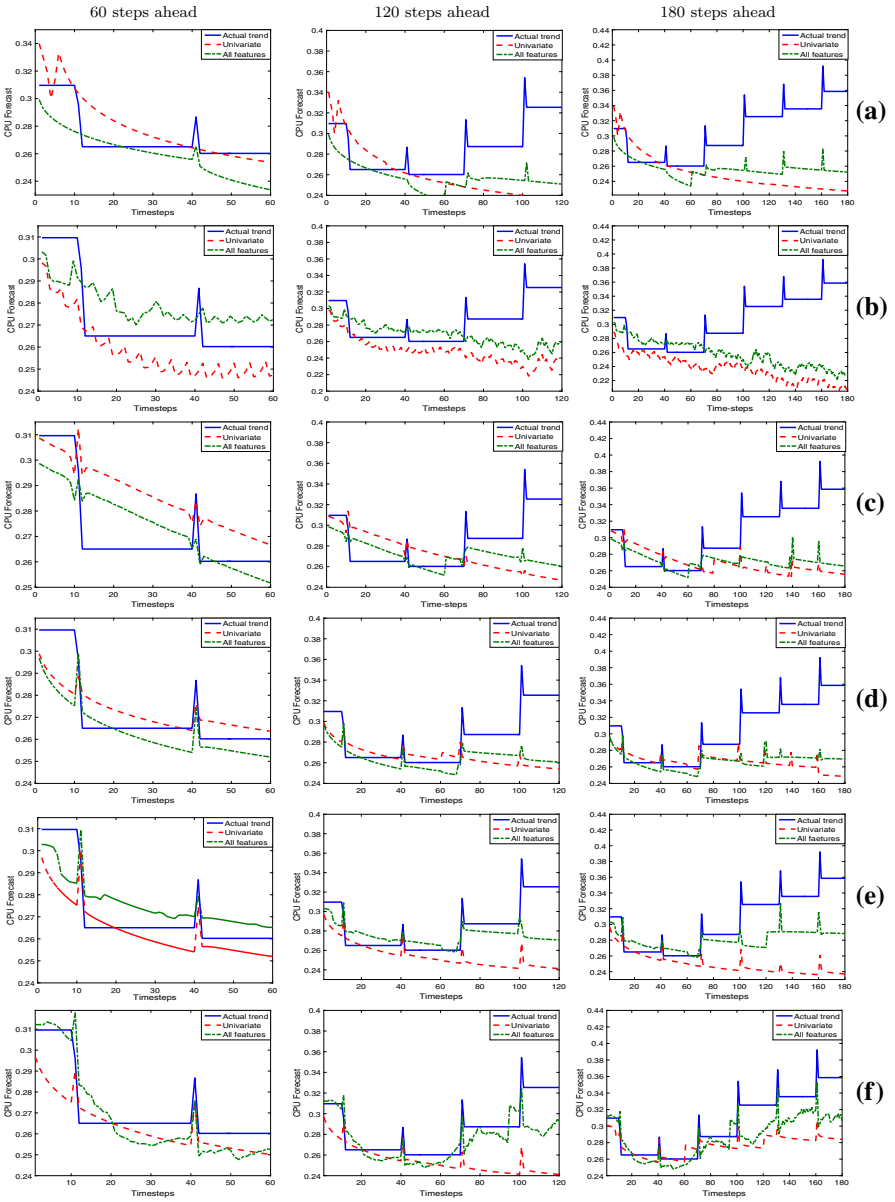


Fig. 8 Multi-step-ahead out-of-sample forecasts using all features in multivariate **a** *f*PRESS-M **b** *f*AGILE-M **c** *f*ARIMA-M **d** *f*NARNN-M **e** LSTM-M **f** BLSTM-M

selected with high weight by all the CPU usage prediction models. This clearly shows the interdependence in the loads of different resources, CPU, memory and disk and hence indicates that better predictions about the CPU usage can be generated by integrating the information about the other resources in joint time series. The prediction

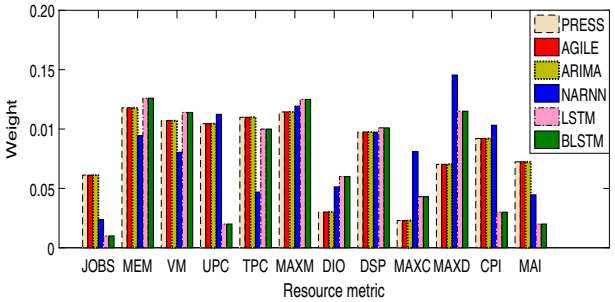


Fig. 9 Analysis of weights given to different features by load prediction models. The x-axis indicates the index of resource metric (features) as given in Table 2

results obtained through the weighted model of time series CPU usage prediction are also shown in Table 3. It is observed that the predictions generated by the weighted combination of all features are marginally better than the predictions obtained using the all features (having equal weights) multivariate models. We observed that in all the cases the BLSTM method performed better and the same can be observed in Fig. 8f.

6.5 CPU usage prediction using feature selection techniques

We now examine the efficacy of our feature selection framework for reducing the set of features. This reduces the space and time complexity of multivariate prediction.

6.5.1 Feature selection using different techniques

Pearson correlation, Spearman correlation, Granger causality and mutual information-based techniques are considered to identify the set of most relevant features from the available set of features.

Figure 10a–d(i) shows the pairwise score Pearson correlation (Fig. 10a(i)), Spearman correlation (Fig. 10b(i)), Granger causality (Fig. 10c(i)) and mutual information (Fig. 10d(i)). Different thresholds are analyzed to identify the significant set of features from the complete set of features. We analyze relevance of features selected at thresholds: $median(M)$, $0.5 * median(0.5 * M)$, $0.75 * median(0.75 * M)$ and $1.5 * median(1.5 * M)$. Here, $median$ is the median of the scores computed between the candidate resource metrics and the desired resource metric. Figure 10a–d(ii) shows the RMSE of different models at different threshold values. From the figure, it is seen that the features that have score greater than median (M) provide the best out-of-sample 180 step ahead prediction. Hence, we have selected only the features that have score higher than median (M) of scores with the CPU usage. Table 4 presents the set of features selected by considering M as threshold, using different feature selection techniques. In the table, a 'Yes' indicates the selection of feature in the technique. It is observed that using Pearson and Spearman correlation methods, the same set of features, viz. disk I/O time (DIO), maximum disk I/O time (MAXD), maximum CPU usage (MAXC), memory usage (MEM) and maximum memory usage (MAXM), are

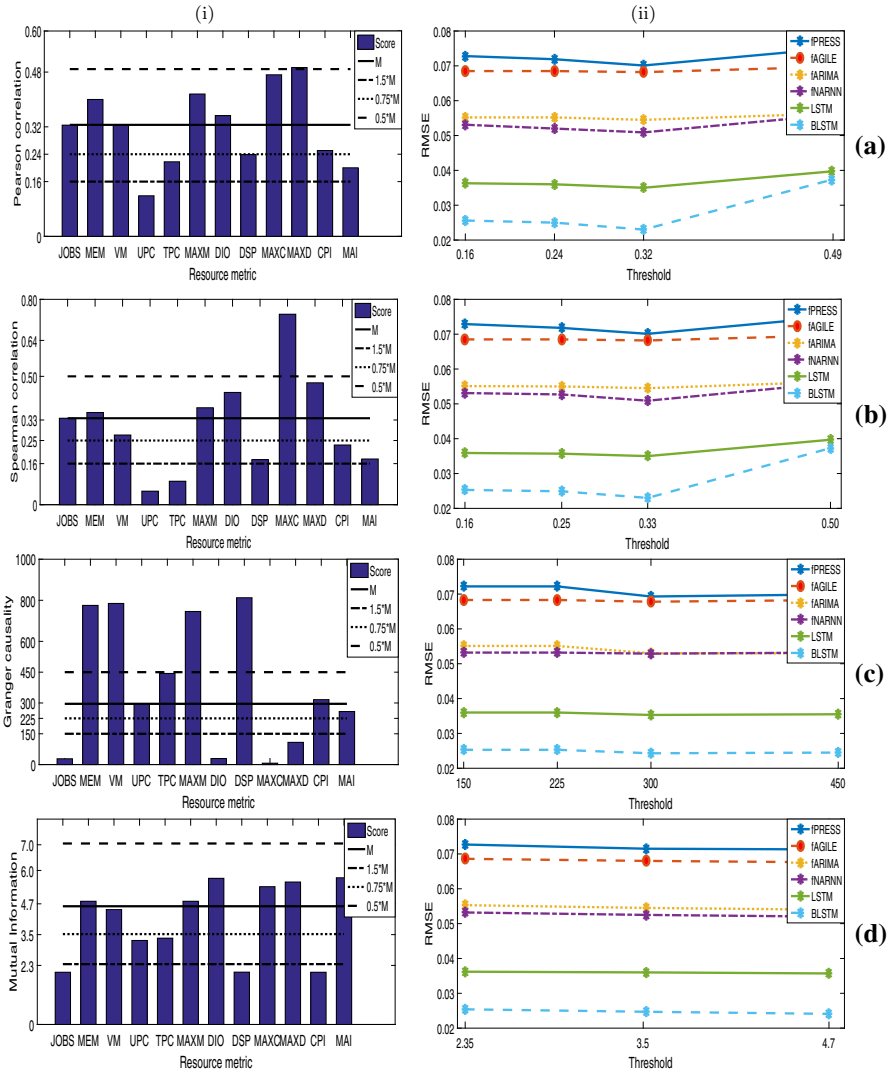


Fig. 10 Plot of different features using **a** Pearson correlation **b** Spearman correlation **c** Granger causality **d** mutual information coefficients. Here the values on x-axis correspond to the resource metrics as indicated in Table 2

selected. Using Granger causality technique, memory usage (MEM), maximum memory usage (MAXM), assigned memory usage (VM), total page cache (TPC), disk space used (DSP) and cycles per instruction (CPI) are chosen as a set of relevant features for load prediction. It is observed that the features selected using mutual information technique came out to be almost same as the features selected using the correlation-based methods. Except that, one extra feature memory accesses per instruction is also a part of set of features selected using mutual information-based method.

Table 4 List of features selected by different feature selection techniques

	JOBS	MEM	VM	UPC	TPC	MAXM	DIO	DSP	MAXC	MAXD	CPI	MAI
Pearson	No	Yes	No	No	No	Yes	Yes	No	Yes	Yes	No	No
Spearman	No	Yes	No	No	No	Yes	Yes	No	Yes	Yes	No	No
Granger causality	No	Yes	Yes	No	Yes	Yes	No	Yes	No	No	Yes	No
Mutual information	No	Yes	No	No	No	Yes	Yes	No	Yes	Yes	No	Yes

6.5.2 Analysis of CPU usage prediction using the selected sets of features

In this subsection, we present the studies on CPU usage prediction using the set of features selected by different feature selection techniques. Table 5 presents the RMSE of the out-of-sample predictions for different multivariate frameworks using set of features selected based on different feature selection techniques. It is observed that multivariate prediction frameworks using selected features perform better than the multivariate prediction frameworks using all features. Different feature selection techniques perform better with different resource usage prediction models. Multivariate PRESS, AGILE and ARIMA models perform better with the features selected using Granger causality method, whereas the neural network models (*f*NARNN, LSTM-M, BLSTM-M) perform better with features selected using correlation- based methods. The RMSE of resource usage prediction models with the features selected using mutual information is comparable to the prediction results of resource prediction models with features selected using correlation-based methods. Figures 11 and 12 show the predictions of load prediction models based on features selected using correlation and causality based methods, respectively, in different multivariate resource usage prediction frameworks. The prediction performance of all frameworks is compared at different prediction horizons of 60 steps ahead, 120 steps ahead and 180 steps ahead. From Fig. 11, it is observed that the multivariate predictions obtained using selected set of features from correlation techniques are better than the predictions by the univariate models. The improvement in the prediction performance of the out-of-sample predictions indicates that selecting the features that are highly correlated with the desired resource metric helps the prediction model in learning the temporal interdependencies between different features, and hence, modeling the variations of related features together generates better predictions than modeling the temporal variations in the desired resource metric alone. From Fig. 11, it is seen that the predictions of all methods degrade as the prediction horizon is increased. This is because of the accumulation of errors while performing multiple steps ahead out-of-sample predictions. However, at 180 step ahead forecast as well, the predictions of multivariate frameworks are better than their corresponding univariate framework. Similarly, it is seen from Fig. 12 that multivariate frameworks with Granger causality based selected features perform better than univariate frameworks. Granger causality analyzes causal relations between different features and hence selects those features that affect the prediction of the desired resource metric. Comparing the prediction performances of different frameworks in Figs. 11 and 12, it is seen that prediction methods generate dif-

Table 5 RMSE of out-of-sample predictions of CPU usage using set of features selected based on different feature selection frameworks

	Correlation-based features			Causality-based features			MI-based features		
	60 steps	120 steps	180 steps	60 steps	120 steps	180 steps	60 steps	120 steps	180 steps
*f*PRESS-M	0.0171	0.0540	0.0701	**0.0169**	**0.0536**	**0.0693**	0.0180	0.0542	0.0713
*f*AGILE-M	0.0126	0.0358	0.0682	**0.0119**	**0.0353**	**0.0678**	0.0129	0.0342	0.0676
*f*ARIMA-M	0.0141	0.0279	0.0545	**0.0131**	**0.0264**	**0.0530**	0.0139	0.0283	0.0540
*f*NARNN-M	**0.0116**	**0.0225**	**0.0509**	0.0124	0.0279	0.0529	0.0118	0.0229	0.0520
LSTM-M	**0.0090**	**0.0190**	**0.0350**	0.0100	0.0193	0.0353	0.0103	0.0197	0.0357
BLSTM-M	**0.0087**	**0.0175**	**0.0238**	0.0089	0.0173	0.0245	0.0092	0.0179	0.0241

Here the entries in bold indicate the feature selection techniques that perform best in different models. MI indicates mutual information

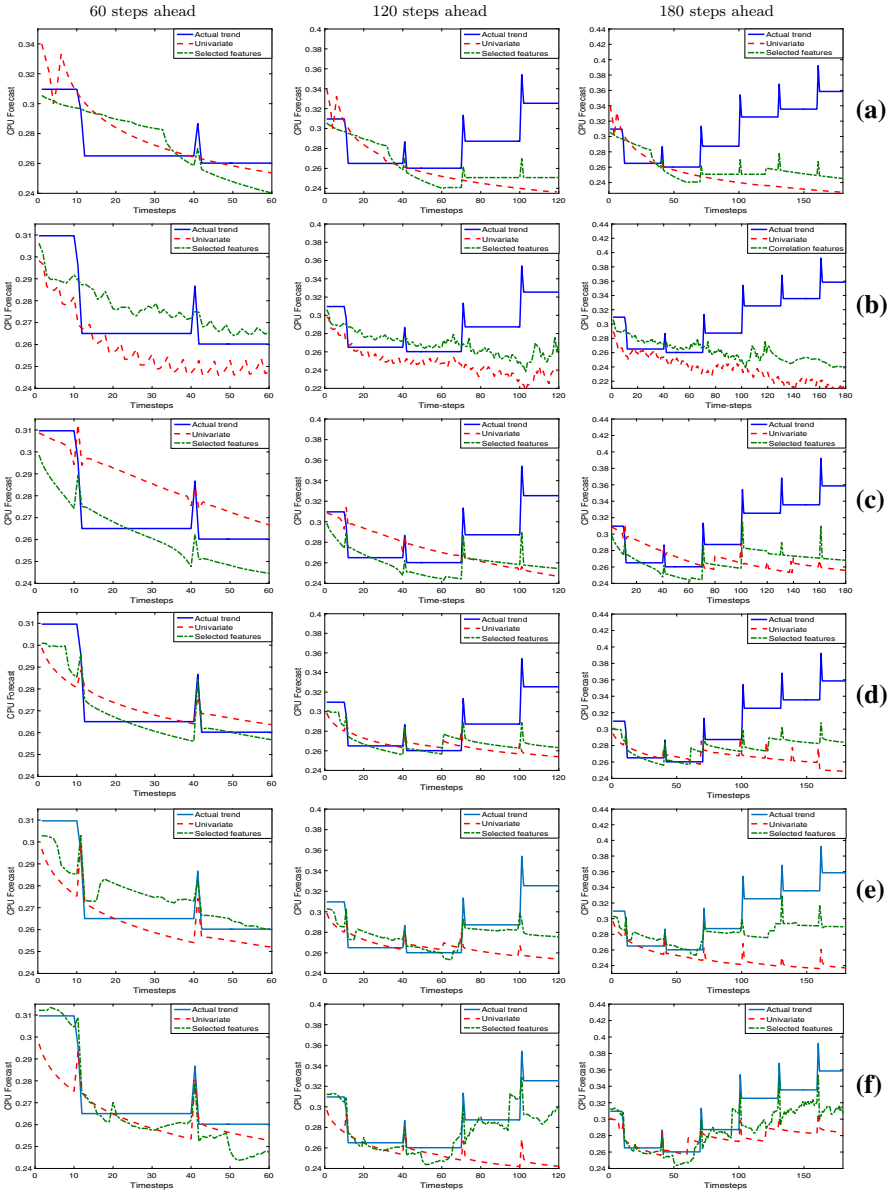


Fig. 11 Multi-step-ahead out-of-sample forecasts with features selected using correlation-based methods in multivariate **a** *f*PRESS-M **b** *f*AGILE-M **c** *f*ARIMA-M **d** *f*NARNN-M **e** LSTM-M **f** BLSTM-M

ferent predictions with different feature selection techniques. For example, comparing Figs. 11c and 12c, it is seen that ARIMA model behaves better with features selected using Granger causality. While comparing Figs. 11d and 12d, it is seen that NARNN learns better temporal variations using the features selected by the correlation method.

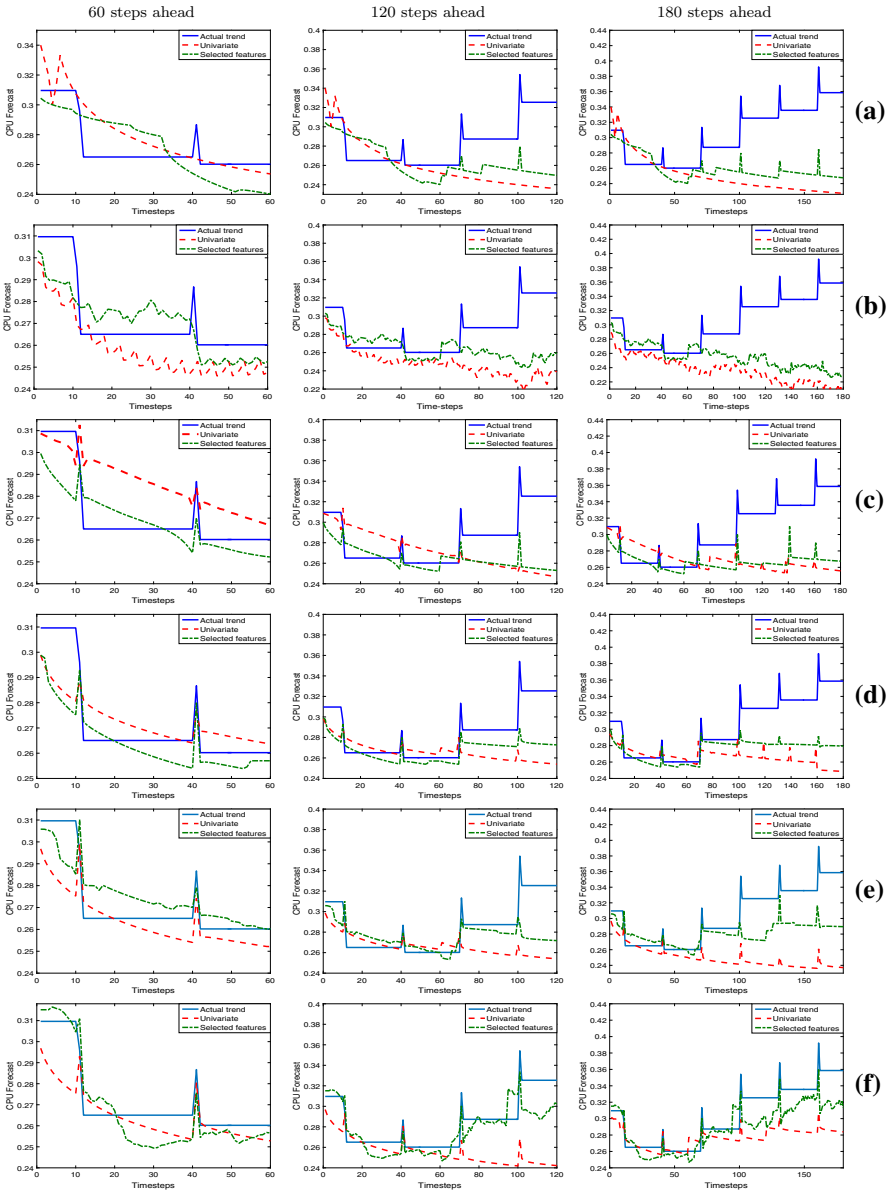


Fig. 12 Multi-step-ahead out-of-sample forecasts with features selected using causality based method in multivariate **a** *f*PRESS-M **b** *f*AGILE-M **c** *f*ARIMA-M **d** *f*NARNN-M **e** LSTM-M **f** BLSTM-M

6.5.3 Extended comparison of feature selection techniques

The resource usage predictions of different models are also analyzed using the set of features that are common in all the feature selection techniques (intersect) and the set of features that are a unification of features selected by all feature selection methods

Table 6 RMSE of out-of-sample predictions of CPU usage using a set of intersect and union features

	Intersect features			Union features		
	60 steps	120 steps	180 steps	60 steps	120 steps	180 steps
*f*PRESS-M	0.0220	0.0553	0.0741	0.0183	0.0538	0.0727
*f*AGILE-M	0.0143	0.0370	0.0694	0.0130	0.0353	0.0680
*f*ARIMA-M	0.0163	0.0304	0.0559	0.0132	0.0280	0.0550
*f*NARNN-M	0.0133	0.0299	0.0547	0.0121	0.0270	0.0529
LSTM-M	0.0115	0.0231	0.0393	0.0103	0.0199	0.0361
BLSTM-M	0.0103	0.0191	0.0269	0.0093	0.0181	0.0235

Here, intersect features indicate the set of features that are chosen by all feature selection techniques and union features indicate a set of features that are union of features selected by all techniques

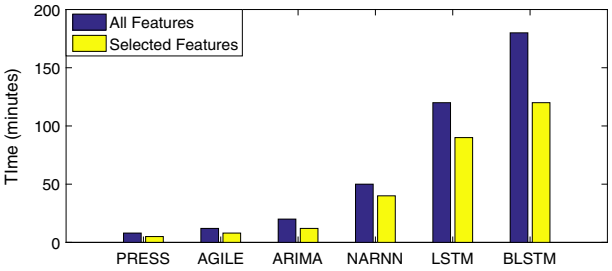


Fig. 13 Comparison of training time of different models build using all features and features selected using Pearson correlation

(union). The set of features that are common (intersect) in all the feature selection techniques are MEM and MAXM. The set of features that are union of all feature selection techniques are MEM, VM, MAXC, MAXM, TPC, CPI, MAI, DIO, MAXD.

Table 6 presents the RMSE of the predictions obtained for the set of intersect and union of features for all the resource usage prediction models. From the table, it is inferred that multivariate models using intersect features generate better predictions than that of the univariate models but are poor as compared to the multivariate prediction models using complete set of features and features selected using different techniques. However, the predictions generated by union set lie in between the predictions generated by different feature selection methods.

The training time of multivariate prediction models built using all features is compared with the models built using the selected features. Figure 13 compares the training time of different models build using all features and set of features selected using Pearson correlation method. It is observed that the models built after selecting an appropriate set of features results in reduced training time as compared to models built from all the features. Therefore, selecting the most appropriate set of features not only results in better predictions but also reduces the time needed to build the prediction models. However, we need to choose appropriate feature selection techniques to select features for better workload predictions. In the next section, we propose to perform stability analysis for choosing appropriate feature selection techniques.

Table 7 Training and test set consistency score of feature selection by different techniques

	Consistency score
Pearson	1.00
Spearman	0.71
Granger causality	1.00
Mutual information	0.71

Table 8 Consistency scores of different feature selection techniques at different overlap ratios with the train set

	0	0.25	0.5	0.75
Pearson	0.68	0.69	0.70	0.71
Spearman	0.68	0.76	0.81	0.77
Granger causality	1.00	1.00	1.00	1.00
Mutual information	0.81	0.86	0.77	0.81

6.5.4 Stability analysis of feature selection techniques

The stability of a feature selection technique is defined as the robustness of the feature subset generated by the technique to variations in the trends in the data. It quantifies the sensitivity of the feature selection technique to changes in the data and supports the selection of a robust feature selection technique [31].

For each feature selection technique, we study the Jaccard similarity of feature selection for training and test sets. The similarity measures the ability of a technique to choose the same set of features for both the training and test set. It is computed using the Jaccard similarity index ϕ [31], as:

$$\phi_{\text{train,test}} = \frac{c_{\text{train,test}}}{k_{\text{train}} + k_{\text{test}} - c_{\text{train,test}}}. \tag{40}$$

Here, $c_{\text{train,test}}$ is the number of features common in feature subsets of two blocks, i.e., train set and test set. k_{train} and k_{test} denote the cardinality of the selected feature subsets train and test set, respectively.

Table 7 presents the similarity score of different feature selection techniques. It is observed that Pearson and Granger causality techniques report the similarity score of 1. This supports the out-of-sample CPU prediction results generated by different feature selection techniques where it is observed that Pearson correlation and Granger causality based feature selection techniques perform best among different multivariate frameworks.

As explained in Sect. 5, we study the stability of feature selection techniques at different overlapping ratios. Table 8 shows the consistency scores of different feature selection techniques at different overlap ratios 0, 0.25, 0.5, 0.75. From the table, it is observed that the consistency scores of Pearson and Spearman correlation-based feature selection techniques are lower as compared to the mutual information and Granger causality based methods. This shows that these techniques are sensitive to the changes in the data. In real-time resource usage predictions, the trends in the

underlying data continuously change with the arrival of new samples. As the set of features selected by these correlation-based techniques are affected by the underlying data, they are not robust and perhaps not the best choice for dynamically changing applications. The consistency scores of the Granger causality and mutual information feature selection techniques are more robust. The scores are high. The consistency score of Granger causality feature selection technique is highest and 1 at different overlap ratios. Hence, it is most robust and consistent in the selection of features.

The joint analysis of performance of feature selection techniques for multivariate resource usage prediction as well as stability of the technique is carried in choosing a better technique for selecting the robust set of features. In Sect. 6.5.2, it is observed that the features selected using correlation and Granger causality based techniques perform best among all for generating out-of-sample CPU load predictions. However, the Granger causality based feature selection technique is more stable than correlation method for selecting the consistent set of features. Therefore, based on their joint analysis, it is observed that in the present context, it will be preferable to use Granger causality for feature selection in the multivariate frameworks for prediction of workload in cloud.

7 Conclusion and future work

Forecasting resource usage is valuable for making better autoscaling and load balancing decisions in cloud servers. The nodes in the cloud can be allocated or deallocated based on the expected future resource usage load on the system. This helps service providers to provide better quality of service to their customers and achieve maximum profits. In this work, multivariate resource usage prediction frameworks are proposed and analyzed for future load prediction. The predictions of different multivariate prediction frameworks are compared with the univariate prediction models, and it is observed that multivariate resource utilization prediction frameworks perform better than the univariate models. Multivariate analysis models temporal variations in more than one resource metric and hence aids in capturing the trends in the usage in a better way. The feature selection techniques based on Pearson correlation, Spearman correlation, Granger causality and mutual information based criterion are analyzed for selection of more relevant subset of features for future CPU usage prediction. As the cloud resource usage workloads are highly time varying in nature, a joint framework based on the prediction performance and stability of feature selection technique are analyzed for selecting the best technique. It is observed that prediction using the set of features selected using Pearson correlation and Granger causality based feature selection techniques is better than the features selected by other techniques. But Granger causality based technique is most stable among all techniques. Based on the joint analysis of the performance of feature selection and stability, it is observed that Granger causality based feature selection technique is well suited for out-of-sample resource usage predictions.

However, cloud workloads are highly time varying in nature. Training the models once using a fixed history of observations may result in more erroneous and uncertain predictions with time. Therefore, this work can be extended to dynamically adapt the

multivariate resource usage prediction model for modeling the changing workloads and generate real-time predictions. Since the usage of resources in cloud depends on the number of users, tasks and performance of different cloud services as well, we will extend our work to analyze the affect of more such factors on cloud resource utilization. Finally, we plan to implement our work framework in an academic cloud testbed to study its performance with real workloads.

References

1. Alelyani S, Zhao Z, Liu H (2011) A dilemma in assessing stability of feature selection algorithms. In: International Conference on High Performance Computing and Communications (HPCC), IEEE, pp 701–707. https://doi.org/10.1109/HPCC.2011.99
2. Borkowski M, Schulte S, Hochreiner C (2016) Predicting cloud resource utilization. In: 9th International Conference on Utility and Cloud Computing (UCC), ACM, New York, USA, pp 37–42. https://doi.org/10.1145/2996890.2996907
3. Caglar F, Gokhale A (2014) iOverbook: intelligent resource-overbooking to support soft real-time applications in the cloud. In: 7th International Conference on Cloud Computing (CLOUD), IEEE, Anchorage, USA, pp 538–545. https://doi.org/10.1109/CLOUD.2014.78
4. Chakraborty K, Mehrotra K, Mohan CK, Ranka S (1992) Forecasting the behavior of multivariate time series using neural networks. Neural Netw 5(6):961–970. https://doi.org/10.1016/S0893-6080(05)80092-9
5. Chen Z, Zhu Y, Di Y, Feng S (2015) Self-adaptive prediction of cloud resource demands using ensemble model and subtractive-fuzzy clustering based fuzzy neural network. Comput Intell Neurosci 919805:17. https://doi.org/10.1155/2015/919805
6. Ching WK, Ng MK, Fung ES (2008) Higher-order multivariate Markov chains and their applications. Linear Algebra Appl 428(23):492–507. https://doi.org/10.1016/j.laa.2007.05.021
7. Dannecker L (2015) Energy time series forecasting: efficient and accurate forecasting of evolving time series from the energy domain, 1st edn. Springer, Berlin. https://doi.org/10.1007/978-3-658-11039-0
8. De Silva AM, Leong PH (2014) Grammar based feature generation for time-series prediction, 1st edn. Springer, Berlin. https://doi.org/10.1007/978-981-287-411-5
9. Di S, Kondo D, Cirne W (2014) Google hostload prediction based on Bayesian model with optimized feature combination. J Parallel Distrib Comput 74(1):1820–1832. https://doi.org/10.1016/j.jpdc.2013.10.001
10. Dougherty B, White J, Schmidt DC (2012) Model-driven auto-scaling of green cloud computing infrastructure. Future Gener Comput Syst 28(2):371–378. https://doi.org/10.1016/j.future.2011.05.009
11. Fang L, Zhao H, Wang P, Yu M, Yan J, Cheng W, Chen P (2015) Feature selection method based on mutual information and class separability for dimension reduction in multidimensional time series for clinical data. Biomed Signal Process Control 21:82–89. https://doi.org/10.1016/j.bspc.2015.05.011
12. Gong Z, Gu X, Wilkes J (2010) PRESS: PRedictive Elastic ReSource Scaling for cloud systems. In: International Conference on Network and Service Management (CNSM), IEEE, Niagara Falls, Canada, pp 9–16. https://doi.org/10.1109/CNSM.2010.5691343
13. Granero MS, Segovia JT, Prez JG (2008) Some comments on hurst exponent and the long memory processes on capital markets. Physica A 387(22):5543–5551. https://doi.org/10.1016/j.physa.2008.05.053
14. Grossglauser M, Bolot JC (1996) On the relevance of long-range dependence in network traffic. IEEE/ACM Trans Netw 26(4):15–24. https://doi.org/10.1109/90.803379
15. Gupta S, Dinesh DA (2017) Resource usage prediction of cloud workloads using deep bidirectional long short term memory networks. In: 11th International Conference on Advanced Networks and Telecommunications Systems (ANTS), IEEE, Bhubaneswar, India, pp 1–6. https://doi.org/10.1109/ANTS.2017.8384098
16. Gupta S, Dileep AD, Gonsalves TA (2016) Fractional difference based hybrid model for resource prediction in cloud network. In: 5th International Conference on Network, Communication and Computing (ICNCC), ACM, Kyoto, Japan, pp 93–97. https://doi.org/10.1145/3033288.3033310

17. Hirwa JS, Cao J (2014) An ensemble multivariate model for resource performance prediction in the cloud. In: Network and Parallel Computing NPC 2014 Lecture Notes in Computer Science, vol 8707, pp 333–346
18. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
19. Hu R, Jiang J, Liu G, Wang L (2013) CPU load prediction using support vector regression and Kalman smoother for cloud. In: 33rd International Conference on Distributed Computing Systems Workshops (ICDCSW), IEEE, Philadelphia, USA, pp 88–92. https://doi.org/10.1109/ICDCSW.2013.60
20. Huang J, Li C, Yu J (2012) Resource prediction based on double exponential smoothing in cloud computing. In: 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), pp 2056–2060. https://doi.org/10.1109/CECNet.2012.6201461
21. Hurst HE (1951) Long-term storage capacity of reservoirs. Trans Am Soc Civ Eng 116:770–808
22. Kaur T, Chana I (2015) Energy efficiency techniques in cloud computing: a survey and taxonomy. ACM Comput Surv 48(2):22:1–22:46. https://doi.org/10.1145/2742488
23. Leland WE, Taqqu MS, Willinger W, Wilson DV (1994) On the self-similar nature of ethernet traffic. IEEE/ACM Trans Netw 2(1):1–15. https://doi.org/10.1109/90.282603
24. Li Z, Wang C, Lv H, Xu T (2015) Research on CPU workload prediction and balancing in cloud environment. Int J Hybrid Inf Technol 8(2):159–172
25. Liang J, Nahrstedt K, Zhou Y (2004) Adaptive multi-resource prediction in distributed resource sharing environment. In: International Symposium on Cluster Computing and the Grid (CCGrid), IEEE, pp 293–300. https://doi.org/10.1109/CCGrid.2004.1336580
26. Liu J, Zhang Y, Zhou Y, Zhang D, Liu H (2015) Aggressive resource provisioning for ensuring QoS in virtualized environments. IEEE Trans Cloud Comput 3(2):119–131. https://doi.org/10.1109/TCC.2014.2353045
27. Liu T, Wei H, Zhang K, Guo W (2016) Mutual information based feature selection for multivariate time series forecasting. In: 35th Chinese Control Conference (CCC), IEEE, Chengdu, China, pp 7110–7114. https://doi.org/10.1109/ChiCC.2016.7554480
28. Mandelbrot BB (1983) The fractal geometry of nature, vol 173. Macmillan, London
29. Messias VR, Estrella JC, Ehlers R, Santana MJ, Santana RC, Reiff-Marganiec S (2016) Combining time series prediction models using genetic algorithm to autoscaling web applications hosted in the cloud infrastructure. Neural Comput Appl 27(8):2383–2406. https://doi.org/10.1007/s00521-015-2133-3
30. Nguyen H, Shen Z, Gu X, Subbiah S, Wilkes J (2013) AGILE: elastic distributed resource scaling for infrastructure-as-a-service. In: 10th International Conference on Autonomic Computing (ICAC), USENIX, San Jose, CA, pp 69–82
31. Nogueira S, Brown G (2016) Measuring the stability of feature selection. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD), Springer, Cham, pp 442–457
32. Peña D, Sánchez I (2007) Measuring the advantages of multivariate vs. univariate forecasts. J Time Ser Anal 28(6):886–909. https://doi.org/10.1111/j.1467-9892.2007.00538.x
33. Reiss C, Wilkes J, Hellerstein JL (2011) Google cluster-usage traces: format + schema. Revised 17 Nov 2014 for version 2.1. Posted at https://github.com/google/cluster-data
34. Shyam GK, Manvi SS (2016) Virtual resource prediction in cloud environment: a Bayesian approach. J Netw Comput Appl 65:144–154. https://doi.org/10.1016/j.jnca.2016.03.002
35. Sims CA (1980) Macroeconomics and reality. Econom J Econom Soc 48(1):1–48. https://doi.org/10.2307/1912017
36. Song B, Yu Y, Zhou Y, Wang Z, Du S (2017) Host load prediction with long short-term memory in cloud computing. J Supercomput. https://doi.org/10.1007/s11227-017-2044-4
37. Sun Y, Li J, Liu J, Chow C, Sun B, Wang R (2015) Using causal discovery for feature selection in multivariate numerical time series. Mach Learn 101(1–3):377–395. https://doi.org/10.1007/s10994-014-5460-1
38. Trapletti A, Leisch F, Hornik K (2000) Stationary and integrated autoregressive neural network processes. Neural Comput 12(10):2427–2450. https://doi.org/10.1162/089976600300015006
39. Wang H, Khoshgoftaar TM, Napolitano A (2015) Stability of three forms of feature selection methods on software engineering data. In: International Conference on Software Engineering and Knowledge Engineering (SEKE), pp 385–390. https://doi.org/10.1142/S0218194015400288

40. Ye J, Xiao C, Esteves RM, Rong C (2015) Time series similarity evaluation based on Spearmans correlation coefficients and distance measures. In: International Conference on Cloud Computing and Big Data in Asia, Springer, pp 319–331
41. Zhang Q, Zhani MF, Zhang S, Zhu Q, Boutaba R, Hellerstein JL (2012) Dynamic energy-aware capacity provisioning for cloud computing environments. In: International Conference on Autonomic Computing (ICAC), ACM, New York, NY, USA, pp 145–154. https://doi.org/10.1145/2371536.2371562
42. Zhang Y, Zhong M, Geng N, Jiang Y (2017) Forecasting electric vehicles sales with univariate and multivariate time series models: the case of China. PLoS ONE 12(5):1–15. https://doi.org/10.1371/journal.pone.0176729