Journal of Network and Systems Management (2020) 28:1775–1793
https://doi.org/10.1007/s10922-020-09557-6

Decomposition Based Cloud Resource Demand Prediction Using Extreme Learning Machines

Jitendra Kumar[1] · **Ashutosh Kumar Singh**[2]

Received: 23 October 2019 / Revised: 9 July 2020 / Accepted: 15 July 2020 / Published online: 31 July 2020

Abstract

Cloud computing has drastically transformed the means of computing in past few years. Apart from numerous advantages, it suffers with a number of issues including resource under-utilization, load balancing and power consumption. The workload prediction is being widely explored to solve these issues using time series analysis regression and neural networks based models. The time series analysis based models are unable to capture the dynamics in the workload behavior whereas neural network based models offer better accuracy on the cost of high training time. This paper presents a workload prediction model based on extreme learning machines (ELM) whose learning time is very low and forecasts the workload more accurately. The performance of the model is evaluated over two real world cloud server workloads i.e. CPU and Memory demand traces of Google cluster and compared with predictive models based on state-of-art techniques including Auto Regressive Integrated Moving Average (ARIMA), Support Vector Regression (SVR), Linear Regression (LR), Differential Evolution (DE), Blackhole Algorithm (BhA), and Propagation (BP). It is observed that the proposed model outperforms the state-of-art techniques by reducing the mean prediction error up to 100% and 99% on CPU and memory request traces respectively.

Keywords Workload forecasting · Google cluster trace · Neural network · Statistical analysis

✉ Jitendra Kumar
jitendra@nitt.edu

Ashutosh Kumar Singh
ashutosh@nitkkr.ac.in

[1] Department of Computer Applications, National Institute of Technology, Tiruchirappalli, Tamilnadu, India

[2] Department of Computer Applications, National Institute of Technology, Kurukshetra, Haryana, India

1 Introduction

With the evolution of cloud computing, resources like processing, memory, and bandwidth are available on-demand over the Internet. The cloud services are enabled with several features such as scalability, mobility, flexibility, elasticity, robustness, and disaster recovery. Elasticity has become a critical feature as it allows an application to scale the resources as per its requirements anytime in its lifespan [1, 2]. However, the rapid changes in the resource demands may force an application to move from one physical machine to another. The resource utilization of a cloud system drops down if the resources are not provisioned efficiently. For instance, IBM observed 17.76% and 77.93% usage of CPU and memory in one of its studies [3]. Similarly, the CPU and memory utilization of Google cluster trance did not exceed 60% and 50% respectively [4]. The electricity consumption grows as the resource utilization drops down due to the fact that more servers will be running than required. The electricity consumption would place the cloud computing sixth in per country ranking. The resource utilization should be improved to reduce the power consumption because an active idle machine consumes over half of the peak power consumption [5]. Moreover, the efficient resource utilization helps in improving the fiscal gain of service provider by reducing the operation cost of the data centers. The resource utilization can be reduced by minimizing the number of active physical machines. Though, finding an optimal mapping of virtual and physical machines in an ever-changing resource requirement scenario is a complex task that belongs to the NP-Complete class of problems [6]. Therefore, an intelligent resource management scheme is required to improve the operational cost of cloud data centers and quality of service (QoS) parameters such as resource availability, elasticity, reliability [7, 8].

Researchers are consistently working towards the development of more effective solutions for resource management. Since workload forecasting allows a system to estimate the resource requirements to fulfill the future demands, it has become one of the essential components of cloud resource management. The prior estimation of resource requirements helps the cloud service providers in providing the uninterrupted services to its consumers. The accuracy of workload forecasting highly affects the decisions of predictive resource scaling models and it is very complex and challenging task to predict accurate cloud workloads due to heavy and dynamic traffic on cloud servers.

The key challenges in precise predictions are interaction with varying number of clients and high non linearity in workload. Machine learning techniques are being explored extensively to propose better forecasting models. Machine learning based approaches use historical data as training window to predict the workload throughout a prediction interval [9]. The advantage of machine learning techniques over statistical methods is that they do not rely on rules given by experts. Instead, they can extract the patterns from data provided to them and give some probabilistic scores on an unknown profile from its experience. The neural network is one of the most popular machine learning approaches that has been used widely in the development of forecasting models. The neural network

based models learn the network weights on the given data using one of the learning algorithms such as backpropagation, genetic algorithm, particle swarm optimization etc. However, these learning algorithms consume a high training time. This paper proposes a predictive model to forecast the cloud server workloads using ELM which is one of the fastest learning algorithms [10–12]. Moreover, the proposed framework decomposes the workload traces into three different components to reduce the presence of non-linearity in the workload.

1.1 Key Contributions

The neural networks have been massively used to develop the predictive models with reasonable accuracy. However, they require extensive amount of training time. The proposed work introduces a predictive model based on neural networks that improves the forecast accuracy and reduces the training time on a large scale. The complex non-linear behavior of real world workload traces is reduced by decomposing them into distinct components that exhibit the simple patterns over real world traces. The model employs an ensemble of neural networks which learns the patterns from distinguished components and predicts the future workload values. Moreover, the ensemble of neural networks is trained using ELM which is one of the fastest learning algorithm as it learns the network weights in single step.

Rest of the paper is organized as follows: Sect. 2 provides an overview of related work. The proposed approach is discussed in Sect. 3 followed by results and discussion in Sect. 4. Finally paper is wrapped up with conclusive remarks and future scope in Sect. 5.

2 Related Work

In general, forecasting finds a wide variety of applications such as stock market price prediction, web service recommendation, and disease prediction [13–15]. It also finds the significant applicability in cloud resource management by estimating the expected workload on the servers [16–18]. It has been observed that machine learning approaches are necessary to deploy prediction models for a large range of applications [19].

2.1 Neural Network Based Approaches

An intelligent workload factoring approach is developed that categorizes the workloads into two categories viz. base crowd and flash crowd based on the different aspects of the applications [20]. A virtual machine prediction scheme is developed for resource provisioning to minimize the electricity consumption of data centers [21]. The scheme also provides an estimation of the required resources to serve the future demands. A predictive framework based on constraint programming and neural network is developed for the dynamic resource provisioning of the cloud resources [22]. A classifier using Bayesian learning is developed to estimate the

workloads of virtual machines [23]. The estimated workload information is used to classify virtual machines as CPU and/or memory intensive workloads to configure the resources accordingly. The neural networks trained using blackhole optimization algorithm are also explored for web and cloud server workload forecasting [24–26]. The cloud server workload is predicted using a fuzzy theory based approach [27]. It forecasts the future resource demands using historical and current CPU utilization. In addition, it also estimates the available resources in near future using the predicted resource utilization. The evolutionary neural network based predictive schemes are introduced in [28–30] to predict the future workload of cloud servers. The networks are trained using adaptive differential evolution learning algorithm that reduces the overhead of parameter tuning. Similarly, a genetic algorithm based workload predictive resource management scheme is introduced to improve the resource usage and power consumption [31]. A comparative study of evolutionary neural network based workload forecasting schemes is carried out [32] that compares the performance of particle swarm optimization, differential evolution, and covariance matrix adaptation evolutionary learning algorithms based predictive models. A study on different servers' resource utilization was carried out in [33] that observed the misalignment of patterns in time. Moreover, a set of algorithms were developed to refine the utilization patterns to reduce the over provisioning for resources. The similar works are reported in [34, 35] that optimize the resource utilization, energy consumption, and secure allocation. The predictive approach involves the use of clustering and Wiener filter. The other similar works are presented in [36–39]. The key limitation of the evolutionary neural network based models is the consumption of high training time [40].

2.2 Deep Learning Based Approaches

A predictive model to estimate the workloads of virtual machines is developed by arranging multiple Boltzmann machines in a layered fashion along with a regression layer [41]. A resource manager composed with monitor, allocator, and controller modules is developed using a deep reinforcement learning algorithm [42]. The monitor, allocator, and controller modules are dedicated to gather the resource utilization information, applications to resource pool mapping, and negotiation of resource configuration correspondingly. The long short term memory (LSTM) recurrent neural network based workload forecasting model is developed to estimate the web server workloads [43]. Similarly, a resource allocation and power management framework is developed using deep learning [44]. The framework includes a forecasting module that forecasts the workload and provides to the power management. The forecasting module is developed using LSTM recurrent neural network. The power manager takes the forecasts and the current state information into account to decide further actions. An efficient workload prediction model based on deep learning is presented in [45] that converts the weight vectors into canonical polyadic decomposition to compress the model attributes. In addition, the work also proposed a learning methodology based on back propagation for the training of the auto encoder's parameters. A prediction model that computes the correlation among virtual machines by

analyzing the past workloads is developed [46]. The deep learning based models offer high accuracy but these approaches encounter with high training time as they need a large number of labeled examples. Moreover, the selection of suitable deep learning architecture is another concern.

2.3 Mining Based Approaches

A cloud resource management approach based on workload forecasting and skewness is presented [47]. The scheme improves the resource utilization by the means of minimizing the skewness. A run-length encoding based forecasting approach is developed to manage the processor power effectively [48]. The approach is energy efficient and effectively addresses the repetitive workloads. A pattern mining based forecasting model is introduced in [49] that detects the correlation between variables and use it to extract the behavioral pattern of applications. The models forecasts the workload information on the servers using extracted patterns. Furthermore, the mining based workload forecasting model with online learning capability is developed [50]. The approach uses two different memories termed as long term memory and short term memory inspired by human memory. The long term memory stores the episodes of application behavior over a long period while the short term memory stores the most recent application behavior that correspond to online learning in the approach.

2.4 Hybrid Approaches

The predictive frameworks that employ only one forecasting model are usually able to fit a specific pattern of workloads and fail in handling the real-world traces where the pattern changes rapidly over time [51]. In such cases the resources remain over and under provisioned. Therefore, a scheme that can adapt to sudden changes becomes more useful. In this context, two online learning ensemble learning approaches for workload prediction are developed [52]. Valter et al. developed a forecasting model that incorporates multiple time series forecasting approaches [53]. In this model, every time series forecasting approach makes its own predictions based on its extracted pattern and these forecasts are weighted to compute the final forecast. The authors used genetic algorithm to generate an effective weighted model. A predictive model that learns and predicts the microservice' workload is presented in [54]. The model uses separate microservices to deploy different components of the prediction model such as training and prediction. The model uses logistic regression and linear regression for multi class classification and regression respectively. The architecture uses predictions for the autoscaling of computing resources of cloud infrastructure. Neural network based predictive model is also explored for 5G core network resource auto scaling [55]. The performance of LSTM and DNN based predictive models is compared and it is observed that forecast based scalability solutions are better than threshold-based solutions for responding to the rapid changes in traffic along with reduction in waiting time to make the resource ready for usage. A workload prediction

scheme based on using weighted random forest was developed in [56] which also introduced an error correction mechanism. The predictive approach employs a set of the random forest where each of them was trained on the different training set. The forecasts of each model were weighted to compute the final forecast. A workload prediction framework 'CloudInsight' was developed using a set of predictors [57]. It combines 8 different prediction methods from machine learning, time series, and regression classes to improve the accuracy of forecasts. The support vector machines were used to predict the workload sequences in [58]. The authors used particle swarm optimization to optimize the model parameters. A number of approaches have been proposed and utilized to forecast the workloads on the servers. It was observed that the above mentioned works were unable to model and forecast the different type of data traces as they were developed and trained for a specific type of workloads. Therefore, the combination of various methods was used to model and forecast the workloads [9, 59]. An ensemble of networks trained using ELM is proposed in [60]. Each network's prediction passes through a weighted voting engine to compute the final forecast. The weight for each network is optimized using one of the metaheuristic algorithms. Similarly, a predictive model is developed to forecast the network virtualization functions workload in cloud computing to effectively allocate the resources for workload execution [61]. It employs an ensemble developed using time series wavelet method and group method in data handling method to forecast the workload and accordingly the workloads are assigned to the physical server. But the existing hybrid forecasting methods suffer from a high computational complexity. On the other hand, the proposed model simplifies the extraction of workload pattern by decomposing the complex workload patterns into multiple and relatively simpler patterns.

The above discussion concludes that most of the predictive frameworks use a single approach or model to anticipate future workload and their accuracy tends to drop down as the pattern of workloads changes. The hybrid approaches have been proposed to address this issue but unfortunately, they suffer from high computational complexity in training. In this paper, a decomposition based forecasting model is proposed to forecast the cloud workloads. The model decomposes the workload trace into three distinct components and trains one network for each of these components. The forecasts of individual networks are combined to get the final forecast. Since the traces are decomposed into simpler components, the model is capable of learning the pattern effectively and forecasts the workload more accurately as shown in the results. The underlying architecture of the network is trained using ELM which is a very fast learning algorithm. The efficacy of the proposed approach is tested on the benchmark datasets however, it can be easily adopted to forecast any data trace.

3 Workload Prediction Approach

A workload predictor is developed using an ensemble of ELMs that analyses the historical data to forecast the resource demands arriving soon on server.

3.1 Extreme Learning Machine

An ELM is essentially a feed forward neural network with one hidden layer which can be used for function approximation. It randomly initializes the weights associated to the synaptic connections between input and hidden neurons which remain constant during training of the network. On the other hand, the weights associated to the synaptic connections between hidden and output neurons are tuned through training. Let $(\boldsymbol{x}_i, y_i)$ be the paired data points where $\boldsymbol{x}_i \in \mathbf{R}^n$ and $y_i \in \mathbf{R}$ are the input and output values of i^{th} data point respectively. Given w_{ih} and w_{ho} are the matrices that store the weights of synaptic connections from input layer to hidden layer and hidden layer to output layer respectively, an ELM that uses K data points for training can be defined as (1).

$$\sum_{i=1}^{L} w_{ho(i)} \times f\left(w_{ih(i)} \cdot \boldsymbol{x}_j + b_i\right) = \hat{y}_j, j = 1, \cdots, K \tag{1}$$

The $w_{ih(i)} \in \mathbf{R}^p$ is a p-dimensional vector that stores the weights of synaptic connections between all input neurons and i^{th} hidden neuron. Similarly, $w_{ih(i)} \cdot \boldsymbol{x}_j$ represents an inner product between $w_{ih(i)}$ and $\boldsymbol{x}_j$, $b_i \in \mathbf{R}$ is the bias weight value connecting bias node and i^{th} hidden node, $f(\cdot)$ is the activation function, $w_{ho(i)} \in \mathbf{R}$ is the weight connecting the i_{th} hidden node to the output node, $\hat{y}_j \in \mathbf{R}$ is the output value of ELM, and L is the number of hidden nodes.

3.2 ELM Based Workload Predictor

The complete workflow of the predictive model is shown in Fig. 1. The resource demand traces are extracted from historical workload data set and then they are aggregated according to a time interval defined as prediction interval window (PWS) which is a time between two consecutive forecasts. Further, difference operator is applied on workload traces to reduce the presence of high non-linearity. The difference operator computes the change between two consecutive workload values. For instance, let $X = \{x_1, x_2, \ldots, x_t\}$ be a series of workload values over time t. The first order difference on the series X can be computed as $\nabla x_t = x_t - x_{t-1}$. Similarly, second order difference on the series X can be obtained as $\nabla(\nabla x_t) = \nabla x_t - \nabla x_{t-1}$. The proposed model uses an ARIMA process to find an optimal order of difference transformation [62]. Next, min-max normalization is used to rescale the workload traces in the range of (0, 1) using $X_{norm} = \frac{X - min}{max - min}$, where X defines the workload

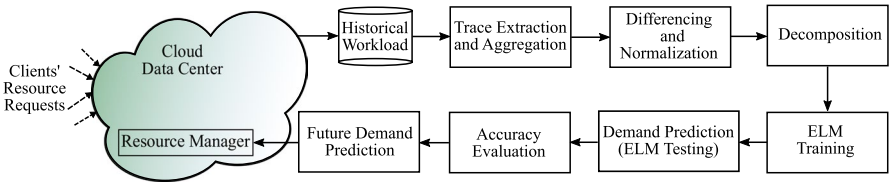


Fig. 1 Workload Prediction Workflow

trace to be rescaled, *min* and *max* are the minimum and maximum of the workload trace respectively.

The preprocessed workload trace is decomposed into three distinct components viz. seasonal, trend, and random. The proposed model uses seasonal decomposition and uses Fourier transforms to detect the seasonality [63]. Since the data characteristics highly affects the choice of decomposition operation and data traces under consideration exhibit no change or low change in the seasonal component over time, additive decomposition is applied on the data traces. The decomposed component traces can be added to reconstruct the original series. If a data trace does not exhibit additive decomposition, an alternative decomposition approach can be employed to extract the simpler data trace components. Since data trace decomposition is an independent process and does not affect the working of predictive model, a new decomposition approach can be integrated easily. An example of original and decomposed traces is shown in Fig. 2.

The prediction model considers all decomposed components as individual traces and uses one neural network for each component. Each network has single output neuron and can be considered as a non-linear function of input data as shown in Eq. (2). The input of the predictive model is a sequence of n recent resource demand values. The performance of a neural network based model depends on various parameters such as number of layers, number of nodes in each layer, activation functions used by different neurons, and synaptic connections among neurons. The proposed predictive model uses three layered neural networks represented as $n-p-q$ structure, where n, p, and q are the number of neurons in input, hidden, and output layer. The output layer uses single neuron as the network has to predict a single value. However, the number of neurons in input and hidden layer should be chosen carefully as they are unknown. The proposed model selects the number of input neurons based on the length of the input pattern, i.e., the number of previous workload instances.

$$x_t = f(x_{t-1}, x_{t-2}, \cdots, x_{t-n}) \tag{2}$$

Since resource demand traces are measured over a regular interval and indexed in time, it can be considered as time series data. The time series data can be modeled to extract a pattern through analyzing historical data. The proposed model makes use of an ARIMA process to find the length of input pattern which is a sequence of n consecutive recent values. An analysis of CPU and memory traces is shown in Table 1a which includes auto regression, integration, and moving average orders. It should be noticed that the transformation order in each case is not more than 5 which indicates that five recent workload values affect the next values most. Thus, the number of input neurons can be selected as 5. However, two other values ($\lfloor 5/2 \rfloor$ and 2×5) of n are selected for experimental purpose. For hidden neurons also, three distinct choices are made randomly. The size of training dataset is another critical parameter that affects the performance of the network. In order to design a list of experiments, three distinct values (50%, 65%, and 85%) are selected randomly. The details of opted values for all parameters are shown in Table 1b. The values of all parameters are shown in Table 1b which generates 27 unique experiment

Table 1 Different factors

(a) ARIMA orders

	O_{AR}	O_I	O_{MA}
CPU	3	1	5
Diff CPU	3	0	5
Memory	1	1	2
Diff memory	1	0	2

(b) Experiment parameters

IN	HN	TSS (%)
2	5	50
5	7	65
10	10	80

configurations. In order to reduce the number of configurations to perform the experiments, a list of 10 experimental configurations are selected using D-optimal design method [64] which are shown in Table 2. Each of the selected configuration is used to perform the experiments to choose the best network structure. The performance of the prediction model is evaluated on a set of unseen input examples using Mean Prediction Error (MPE) [17, 65, 66] which computes the deviation of the predictions from actual workload on the server. A detailed analysis of the prediction model with different configurations is given in next section. The trained predictive model can be deployed in the cloud system to forecast the cloud resource demands before actual demands arrive. The predicted resource demands can be fed into the resource manager of the cloud data center which can be effectively used in resource scaling decisions. A cloud system may use the predicted resource demands to increase the resource utilization and their availability provided that the predictions are reasonably accurate.

4 Results and Discussion

The experiments are conducted on a machine equipped with main memory of 6 GB and dual Intel Core i5-3230M processors running at 2.60 GHz. A set of experiments are conducted on CPU and memory demands of Google cluster trace [4]. The Google cluster trace is a collection of 29 days' observations of 7000 servers running as a cluster. The dataset is a record of 672075 jobs and more than 48 million tasks running on Google cluster. The experiments are conducted for time intervals of 1, 10, 20, 30, 60 min, 1 day.

Table 2 Experiments selected by D-optimal experiment design

Exp no	Input node	Hidden node	Training data size (%)
1	2	10	50
2	5	5	50
3	10	10	50
4	2	5	80
5	5	10	65
6	10	7	65
7	2	10	80
8	5	7	80
9	10	5	80
10	10	10	80

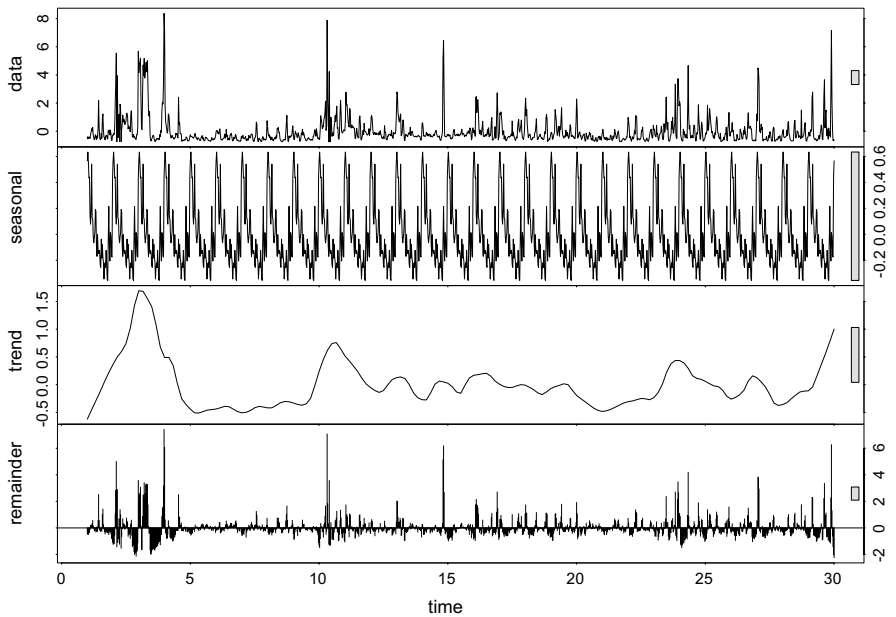


Fig. 2 Decomposition of CPU request

4.1 Forecast Results

Each experiment is repeated 1000 times for different system configurations and average of the results is reported in this study to generalize the results. The forecast accuracy is measured using mean prediction error (MPE) [17, 66] that can be defined as given in Eq. (3), where $\hat{y}_i$ and y_i are the predicted and actual values

of resource demands respectively, and m is the number of training patterns under consideration.

$$MPE = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2 \tag{3}$$

The forecast error on CPU trace for different system configurations selected using design of experiments are listed in Table 3. Since an ideal prediction model should predict the workload with no error, the objective is to minimize the forecast error. The minimum forecast error on CPU trace is generated by a prediction model configured with 2 input neurons and 10 hidden neurons as highlighted in Table 3. Similarly, Table 4 lists the forecast errors obtained by the proposed model configured with different parameters on memory demand trace. In this case, a model with 5 input neurons and 10 hidden neurons achieved the least forecast error. It uses 65% of data to learn the synaptic connection weights effectively. It should be noted that forecast accuracy of predictive models is different on CPU and memory trace. The data characteristics including the training data size highly affect the forecast accuracy of a predictive model based on neural network. The CPU trace and memory trace are two different traces and a neural network trained on one dataset may not be expressive enough to model another data as highlighted in [67]. The actual and predicted values for the resource demand traces are pictorially shown in Fig. 3. In order to improve the visibility of the graphs, a set of 20 consecutive data points are randomly selected from the data traces which are pictorially shown along with their forecast values. The visuals show that the forecasts are very close to the actual values. The forecasts with higher proximity to their actual values can be effectively used by the resource manager in keeping the quality of service high along with a substantial avoidance of service level agreement violations.

The performance of the proposed model is compared with multiple state-of-art techniques based predictive models including adaptive differential evolution (SaDE), blackhole algorithm (BhA), back-propagation (BPNN), support vector regression (SVR), linear regression (LR), and auto regression integrated moving average (ARIMA). Table 5 lists the comparison of the models on CPU trace forecasts. The relative reduction in the forecast error is computed as $\frac{E_{SA} - E_{PR}}{E_{SA}} * 100$, where E_{SA} and E_{PR} are the forecast error of state-of-art model and proposed model respectively. It is observed that the proposed model substantially reduces the forecast errors with a relative reduction close to 85%, 79%, 98%, 100%, 100%, and 100% over state-of-art models based on BhA, SaDE, BPNN, ARIMA, SVR, and LR respectively. Similarly, the forecast errors on memory trace are compared and the results are listed in the Table 6. The proposed model observed the relative reduction in the forecast error close to 49%, 41%, 92%, 99%, 99%, and 99% over state-of-art models based on BhA, SaDE, BPNN, ARIMA, SVR, and LR respectively. Based on the results it can be observed that proposed model outperforms the state-of-art forecasting approaches. Moreover, the proposed model extensively reduces the running time as compared to the existing solutions as shown in

Table 3 Proposed models's CPU trace forecasting errors for different experiment configurations

Exp no	Prediction interval					
	1 min	10 min	20 min	30 min	60 min	1 day
E−1	3.07E−03	3.04E−03	3.05E−03	2.98E−03	3.09E−03	3.01E−03
E−2	3.40E−03	3.42E−03	3.42E−03	3.39E−03	3.44E−03	3.40E−03
E−3	3.32E−03	3.34E−03	3.34E−03	3.32E−03	3.34E−03	3.33E−03
E−4	2.27E−03	2.25E−03	2.27E−03	2.26E−03	2.27E−03	2.25E−03
E−5	2.80E−03	2.81E−03	2.82E−03	2.82E−03	2.82E−03	2.81E−03
E−6	3.42E−03	3.43E−03	3.44E−03	3.41E−03	3.44E−03	3.46E−03
E−7	**2.17E−03**	**2.20E−03**	**2.17E−03**	**2.16E−03**	**2.16E−03**	**2.17E−03**
E−8	2.33E−03	2.32E−03	2.33E−03	2.32E−03	2.33E−03	2.32E−03
E−9	2.71E−03	2.70E−03	2.70E−03	2.71E−03	2.71E−03	2.69E−03
E−10	2.44E−03	2.45E−03	2.42E−03	2.44E−03	2.43E−03	2.43E−03

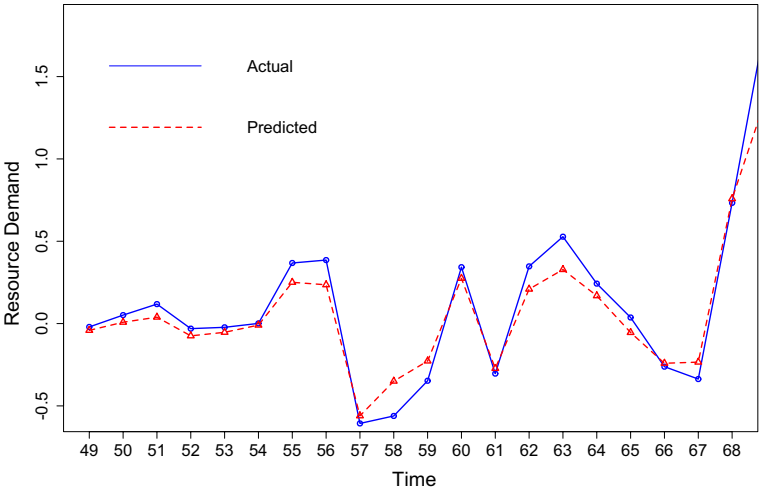
Table 4 Proposed models's Memory trace forecasting errors for different experiment configurations

Exp no	Prediction interval					
	1 min	10 min	20 min	30 min	60 min	1 day
E−1	7.27E−03	7.25E−03	7.28E−03	7.28E−03	7.29E−03	7.27E−03
E−2	7.52E−03	7.49E−03	7.52E−03	7.61E−03	7.57E−03	7.54E−03
E−3	7.47E−03	7.47E−03	7.45E−03	7.45E−03	7.47E−03	7.48E−03
E−4	7.60E−03	7.59E−03	7.64E−03	7.63E−03	7.61E−03	7.62E−03
E−5	**6.97E−03**	**6.95E−03**	**6.98E−03**	**6.98E−03**	**6.95E−03**	**6.98E−03**
E−6	7.81E−03	7.88E−03	7.80E−03	7.84E−03	7.80E−03	7.77E−03
E−7	7.66E−03	7.79E−03	7.64E−03	7.68E−03	7.77E−03	7.62E−03
E−8	7.64E−03	7.62E−03	7.60E−03	7.61E−03	7.67E−03	7.64E−03
E−9	8.77E−03	8.75E−03	8.72E−03	8.74E−03	8.66E−03	8.72E−03
E−10	8.08E−03	8.03E−03	8.03E−03	8.07E−03	8.05E−03	8.04E−03

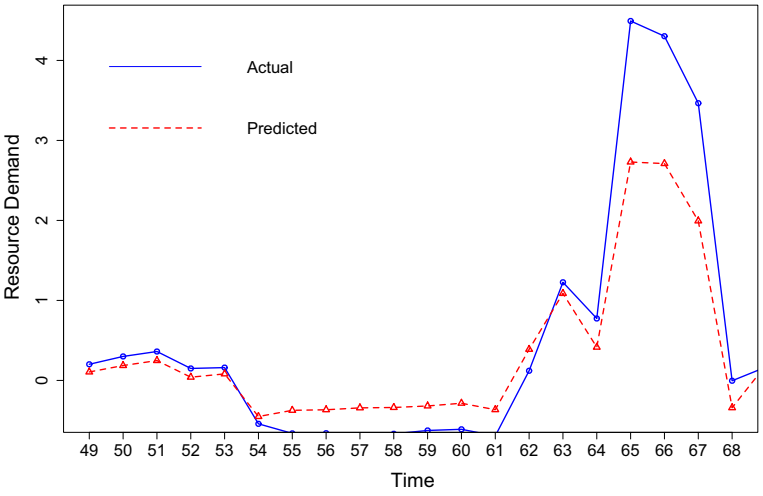
Tables 7 and 8 for CPU and memory trace respectively. The training time is reduced from 3 times to $2.6E+05$ times.

4.2 Statistical Analysis and Discussion

The statistical analysis is conducted using Friedman ranking and Finner post-hoc analysis tests to validate the efficacy of the forecast results [68, 69]. The Friedman test follows a null hypothesis ($H_{0_{\text{fr}}}$) which assumes that the results of multiple algorithms are same. The performance of the algorithms differs as the null hypothesis is rejected. The rank of each algorithm obtained from the Friedman test is shown in Table 9. The proposed model achieved the same rank as the



(a) CPU Trace



(b) Memory Trace

Fig. 3 Forecast of resource requests (uniformly selected 20 points from testing data)

Table 5 Performance comparison of different models on CPU trace forecasting

PWS	Proposed	SADE	BHA	BPNN	ARIMA	SVR	LR
1	2.17E−03	8.48E−04	9.26E−04	4.41E−03	5.94E−01	1.05E+00	1.03E+00
10	2.20E−03	2.47E−03	3.59E−03	1.42E−02	9.98E−01	7.16E−01	7.18E−01
20	2.17E−03	4.02E−03	6.28E−03	2.60E−02	9.96E−01	6.04E−01	6.29E−01
30	2.16E−03	5.49E−03	1.13E−02	4.37E−02	8.27E−01	5.34E−01	5.66E−01
60	2.16E−03	1.03E−02	1.43E−02	1.03E−01	8.67E−01	6.91E−01	6.89E−01

self-adaptive differential evolution learning algorithm based forecasting model. However, the proposed solution outperforms the latter solution in training time. The predictive models based on blackhole algorithm, back-propagation algorithm, SVR, LR, and ARIMA are ranked afterwards. The Finner post-hoc analysis is conducted to evaluate the performance of proposed model against other models [69]. The pairwise tests are conducted around a null hypothesis ($H_{0_{\text{fn}}}$) that assumes the similarity in the performance of the paired algorithms. Table 10 lists out the detailed observations obtained from the test. The test confirms the statistical in the forecasts of ARIMA, LR, SVR, and BPNN based predictive networks by rejecting the null hypothesis of Finner test. Further, the test accepts the null hypothesis against blackhole and self adaptive differential evolution learning algorithm based models. The tests confirm that the performance of these algorithms are statistically the same. However, the proposed model learns the synaptic connection weights in single step which gives a very short training time in comparison to iterative based algorithm. Therefore, the proposed solution is considered better than other approaches.

Table 6 Performance comparison of different models on Memory trace forecasting

PWS	Proposed	SADE	BhA	BPNN	ARIMA	SVR	LR
1	6.97E−03	2.35E−04	2.48E−04	1.97E−03	5.06E−01	1.36E+00	1.23E+00
10	6.96E−03	4.08E−03	8.56E−03	2.48E−02	8.89E−01	9.72E−01	9.94E−01
20	6.98E−03	5.54E−03	1.36E−02	4.80E−02	5.81E−01	5.18E−01	5.23E−01
30	6.99E−03	9.51E−03	1.15E−02	5.16E−02	5.91E−01	5.27E−01	5.30E−01
60	6.96E−03	1.18E−02	1.09E−02	8.77E−02	6.10E−01	7.57E−01	7.55E−01

Table 7 Comparison of training time (s) on CPU trace

PWS	Proposed	SaDE	BhA	BPNN	ARIMA	SVR	LR
1	0.009	979.151	2511.657	803.954	53.610	221.020	0.638
10	0.009	47.540	33.311	73.816	7.860	2.060	0.089
20	0.009	22.858	126.084	37.965	0.500	0.550	0.062
30	0.009	14.741	85.025	25.006	0.270	0.250	0.054
60	0.009	11.423	42.511	13.166	0.200	0.140	0.033

Table 8 Comparison of training time (sec) on memory trace

PWS	Proposed	SaDE	BhA	BPNN	ARIMA	SVR	LR
1	0.010	634.830	2450.621	700.860	22.91	337.27	0.635
10	0.009	84.321	236.560	81.802	1.75	5.48	0.092
20	0.009	22.552	13.622	37.071	1.46	1.12	0.062
30	0.009	18.425	78.145	25.012	0.23	0.64	0.054
60	0.009	6.684	39.151	10.409	0.48	0.45	0.035

Table 9 Friedman test rankings

Algorithm	Ranking
ELM	1.7
SaDE	1.7
BhA	2.7
BPNN	3.9
SVR	5.8
LR	6.0
ARIMA	6.2

Table 10 Post hoc analysis using Finner test

Comparison	Statistic	Adjusted p-value	Result
ELM vs ARIMA	4.65794	0.00002	$H_{0_{\text{fn}}}$ is rejected
ELM vs LR	4.45092	0.00003	$H_{0_{\text{fn}}}$ is rejected
ELM vs SVR	4.24390	0.00004	$H_{0_{\text{fn}}}$ is rejected
ELM vs BPNN	2.27722	0.03396	$H_{0_{\text{fn}}}$ is rejected
ELM vs BHA	1.03510	0.34889	$H_{0_{\text{fn}}}$ is accepted
ELM vs SADE	0.00000	1.00000	$H_{0_{\text{fn}}}$ is accepted

5 Conclusions

The predictive cloud resource management frameworks are effectively being used to improve the various parameters of service oriented paradigm such as resource utilization, energy consumption, QoS, SLA violations, operational cost etc. However, the improvement in these parameters highly depends on the accuracy of the forecasts. The proposed model improves the forecast accuracy and learns the network weights much faster than existing solutions. The proposed model decomposes the complex patterns of the workload traces into distinct and simple components. An ensemble of neural networks is created to learn the patterns from each extracted components. The performance of the proposed model is evaluated on real workload traces of Google cluster traces and compared with state of art prediction models. The experimental observations are convincing and outperform the existing solutions. The proposed approach has relatively reduced the forecast mean prediction error from 11 to 100%. It also reduces the training time by a large factor. Thus, the proposed model can also improve the other factors of the cloud system such as resource utilization, SLA violations, operational cost etc.

However, the proposed predictive model has two major limitations i.e. it forecasts only single variable and it does not optimizes the network structure hyper-parameters such as number of hidden layers. This work can be extended to modify the network structure to forecast multiple variables. Similarly, an automatic structure learning algorithm can be integrated in the proposed approach to make it self optimized network structure model.

Acknowledgements This research was supported by funding from Ministry of Electronics and Information Technology (MeitY), Government of India.

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Pandey, S., Sammut, L., Calheiros, R.N., Melatos, A., Buyya, R.: Scalable deployment of a LIGO physics application on public clouds: workflow engine and resource provisioning techniques. pp. 3–25
2. Assunção, M.D.D., Veith, A.D.S., Buyya, R.: Resource elasticity for distributed data stream processing: a survey and future directions. CoRR abs/1709.01363 (2017)
3. Birke, R., Chen, L.Y., Smirni, E., Birke, R., Chen, L.Y., Smirni, E.: Data centers in the wild: a large performance study. In: Tech. rep, IBM Research—Zurich, Switzerland (2012)
4. Reiss, C., Tumanov, A., Ganger, G.R., Katz, R.H., Kozuch, M.A.: Heterogeneity and dynamicity of clouds at scale: Google trace analysis. ACM Symp. Cloud Comput. **2012**, 1–18 (2012)
5. Barroso, L., Hölzle, U.: The datacenter as a computer an introduction to the design of Warehouse-Scale Machines, vol. 24. Morgan & Claypool Publishers, New York (2013)
6. Li, X., Qian, Z., Lu, S., Wu, J.: Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center. Math. Comput. Model. **58**(5), 1222–1235 (2013)
7. Yousif, M.: The state of the cloud. IEEE Cloud Comput. **5**(1), 4–5 (2018)
8. Kumar, J., Singh, A.K.: Cloud datacenter workload estimation using error preventive time series forecasting models. Cluster Comput. **23**, 1363–1379 (2020)
9. Cetinski, K., Juric, M.B.: AME-WPC: Advanced model for efficient workload prediction in the cloud. Journal of Network and Computer Applications **55**, 191–201 (2015)
10. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: theory and applications. Neurocomputing **70**, 489–501 (2006)
11. Huang, G., Huang, G.B., Song, S., You, K.: Trends in extreme learning machines: a review. Neural Netw. **61**, 32–48 (2015)
12. Huang, G., Song, S., Gupta, J.N.D., Wu, C.: Semi-supervised and unsupervised extreme learning machines. IEEE Trans. Cybern. **44**(12), 2405–2417 (2014)
13. Singh, S., Madan, T.K., Kumar, J., Singh, A.K.: Stock market forecasting using machine learning: Today and tomorrow. In: 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT). vol. 1, pp. 738–745 (2019)
14. Su, K., Xiao, B., Liu, B., Zhang, H., Zhang, Z.: TAP: a personalized trust-aware qos prediction approach for web service recommendation. Knowl. Based Syst. **115**, 55–65 (2017)
15. Sharma, V., Kaur, S., Kumar, J., Singh, A.K.: A fast parkinson's disease prediction technique using PCA and artificial neural network. In: 2019 International Conference on Intelligent Computing and Control Systems (ICCS). pp. 1491–1496 (2019)
16. Jennings, B., Stadler, R.: Resource management in clouds: survey and research challenges. J. Netw. Syst. Manag. **23**(3), 567–619 (2015)
17. Bi, J., Yuan, H., Zhang, L., Zhang, J.: SGW-SCN: an integrated machine learning approach for workload forecasting in geo-distributed cloud data centers. Inform. Sci. **481**, 57–68 (2019)
18. Yin, J., Lu, X., Chen, H., Zhao, X., Xiong, N.N.: System resource utilization analysis and prediction for cloud based applications under bursty workloads. Inform. Sci. **279**, 338–357 (2014)
19. Witt, C., Bux, M., Gusew, W., Leser, U.: Predictive performance modeling for distributed batch processing using black box monitoring and machine learning. Inform. Syst. **82**, 33–52 (2019)
20. Zhang, H., Jiang, G., Yoshihira, K., Chen, H.: Proactive workload management in hybrid cloud computing. IEEE Trans. Netw. Service Manag **11**(1), 90–100 (2014)
21. Dabbagh, M., Hamdaoui, B., Guizani, M., Rayes, A.: Energy-efficient resource allocation and provisioning framework for Cloud Data Centers. IEEE Trans. Netw. Service Manag **12**(3), 377–391 (2015)

22. Wamba, G.M., Li, Y., Orgerie, A.C., Beldiceanu, N., Menaud, J.M.: Cloud workload prediction and generation models. In: Proceedings-29th International Symposium on Computer Architecture and High Performance Computing, SBAC-PAD 2017 pp. 89–96 (2017)
23. Kumaraswamy, S., Nair, M.K.: Intelligent VMs prediction in cloud computing environment. In: 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon) pp. 288–294 (2017)
24. Kumar, J., Singh, A.K.: Dynamic resource scaling in cloud using neural network and black hole algorithm. In: 2016 Fifth International Conference on Eco-friendly Computing and Communication Systems (ICECCS). pp. 63–67 (2016)
25. Kumar, J., Singh, A.K.: An efficient machine learning approach for virtual machine resource demand prediction. Int. J. Adv. Sci. Technol. **123**, 21–30 (2019)
26. Kumar, J., Singh, A.K., Buyya, R.: Self directed learning based workload forecasting model for cloud resource management. Inf. Sci. (2020)
27. Ramezani, F.: A fuzzy virtual machine workload prediction method for cloud environments. In: 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). pp. 1–6 (2017)
28. Kumar, J., Singh, A.K.: Workload prediction in cloud using artificial neural network and adaptive differential evolution. Future Gen. Comput. Syst. **81**, 41–52 (2018)
29. Kumar, J., Saxena, D., Singh, A.K., Mohan, A.: Biphase adaptive learning based neural network model for cloud workload forecasting. Soft Comput. (2020)
30. Kumar, J., Singh, A.K.: Adaptive learning based prediction framework for cloud datacenter networks' workload anticipation. J. Inf. Sci. Eng. (2020)
31. Tseng, F.H., Wang, X., Chou, L.D., Chao, H.C., Leung, V.C.M.: Dynamic resource prediction and allocation for Cloud Data Center using the multiobjective genetic algorithm. IEEE Syst. J. **12**(2), 1688–1699 (2018)
32. Mason, K., Duggan, M., Barrett, E., Duggan, J., Howley, E.: Predicting host CPU utilization in the cloud using evolutionary neural networks. Future Gen. Comput. Syst. **86**, 162–173 (2018)
33. Shen, H., Chen, L.: Resource demand misalignment: an important factor to consider for reducing resource over-provisioning in cloud datacenters. IEEE/ACM Trans. Netw. **10**, 1–15 (2018)
34. Singh, A.K., Kumar, J.: Secure and energy aware load balancing framework for cloud data centre networks. Elect. Lett. **55**(1), 540–541 (2019)
35. Kumar, J., Singh, A.K., Mohan, A.: Resource-efficient load-balancing framework for cloud data center networks. ETRI J. (2020). https://doi.org/10.4218/etrij.2019-0294
36. Kumar, J., Singh, A.K.: Cloud resource demand prediction using differential evolution based learning. In: 2019 7th International Conference on Smart Computing Communications (ICSCC). pp. 1–5 (2019)
37. Prevost, J.J., Nagothu, K., Kelley, B., Jamshidi, M.: Prediction of cloud data center networks loads using stochastic and neural models. In: 2011 6th International Conference on System of Systems Engineering. pp. 276–281 (2011)
38. Chang, Y.C., Chang, R.S., Chuang, F.W.: A predictive method for workload forecasting in the cloud environment, pp. 577–585. Springer, Dordrecht (2014)
39. Lu, Y., Panneerselvam, J., Liu, L., Wu, Y.: RVLBPNN: a workload forecasting model for smart cloud computing. Sci. Prog. **2016**, 1–9 (2016)
40. Amiri, M., Mohammad-Khanli, L.: Survey on prediction models of applications for resources provisioning in cloud. J. Netw. Comput. Appl. **82**, 93–113 (2017)
41. Qiu, F., Zhang, B., Guo, J.: A deep learning approach for VM workload prediction in the cloud. In: 2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD). pp. 319–324 (2016)
42. Zhang, Y., Yao, J., Guan, H.: Intelligent cloud resource management with deep reinforcement learning. IEEE Cloud Computing **4**(6), 60–69 (2017)
43. Kumar, J., Goomer, R., Singh, A.K.: Long short term memory recurrent neural network (LSTM-RNN) based workload forecasting model for Cloud Datacenters. Procedia Comput. Sci. **125**, 676–682 (2018)
44. Liu, N., Li, Z., Xu, J., Xu, Z., Lin, S., Qiu, Q., Tang, J., Wang, Y.: A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning. In: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). pp. 372–382 (2017)
45. Zhang, Q., Yang, L.T., Yan, Z., Chen, Z., Li, P.: An efficient deep learning model to predict cloud workload for industry informatics. IEEE Trans. Ind. Inf. **20**, 1–9 (2018)

46. Patel, Y.S., Misra, R.: Performance comparison of deep VM workload prediction approaches for Cloud. In: Pattnaik, P., Rautaray, S., Das, H., Nayak, J. (eds.) Progress in Computing, Analytics and Networking. Advances in Intelligent Systems and Computing, pp. 149–160. Springer, Singapore (2018)
47. Xiao, Z., Member, S., Song, W., Chen, Q.: Dynamic resource allocation using virtual machines for cloud computing environment. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1107–1117 (2013)
48. Kim, S., Kim, T., Yoo, C.: Workload prediction using run-length encoding for runtime processor power management. Elect. Lett. **51**(22), 1759–1761 (2015)
49. Amiri, M., Mohammad-Khanli, L., Mirandola, R.: A sequential pattern mining model for application workload prediction in cloud environment. J. Netw. Comput. Appl. **105**, 21–62 (2018)
50. Amiri, M., Mohammad-Khanli, L., Mirandola, R.: An online learning model based on episode mining for workload prediction in cloud. Future Gen. Comput. Syst. **87**, 83–101 (2018)
51. Chen, Z., Zhu, Y., Di, Y., Feng, S.: Self-adaptive prediction of cloud resource demands using ensemble model and subtractive-fuzzy clustering based fuzzy neural network. Comput. Intell. Neurosci. **2015**, 14 (2015)
52. Singh, N., Rao, S.: Ensemble learning for large-scale workload prediction. IEEE Trans. Emerg. Topics Comput. **2**(2), 149–165 (2014)
53. Messias, V.R., Estrella, J.C., Ehlers, R., Santana, M.J., Santana, R.C., Reiff-Marganiec, S.: Combining time series prediction models using genetic algorithm to autoscaling web applications hosted in the cloud infrastructure. Neural Comput. Appl. **27**(8), 2383–2406 (2016)
54. Alipour, H., Liu, Y.: Online machine learning for cloud resource provisioning of microservice backend systems. In: 2017 IEEE International Conference on Big Data (Big Data). pp. 2433–2441 (2017)
55. Alawe, I., Ksentini, A., Hadjadj-Aoul, Y., Bertin, P.: Improving traffic forecasting for 5g core network scalability: a machine learning approach. IEEE Netw. **32**(6), 42–49 (2018)
56. Chen, M., Yuan, J., Liu, D., Li, T.: An adaption scheduling based on dynamic weighted random forests for load demand forecasting. J. Supercomput. **12**, 1–19 (2017)
57. Kim, I.K., Wang, W., Qi, Y., Humphrey, M.: CloudInsight: utilizing a council of experts to predict future cloud application workloads. In: In 10th IEEE International Conference on Cloud Computing (Cloud 2018), July 2 - July 7. pp. 1–8. San Francisco, USA (2018)
58. Zhong, W., Zhuang, Y., Sun, J., Gu, J.: A load prediction model for cloud computing using PSO-based weighted wavelet support vector machine. Appl. Intell. 20, 1–12 (2018)
59. Jiang, Y., Perng, C., Li, T., Chang, R.N.: Cloud analytics for capacity planning and instant VM provisioning. IEEE Trans. Netw. Service Manag. **10**(3), 312–325 (2013)
60. Kumar, J., Singh, A.K., Buyya, R.: Ensemble learning based predictive framework for virtual machine resource request prediction. Neurocomputing **397**, 20–30 (2020)
61. Jeddi, S., Sharifian, S.: A hybrid wavelet decomposer and gmdh-elm ensemble model for network function virtualization workload forecasting in cloud computing. Appl. Soft Comput. **88**, 105940 (2020)
62. Hyndman, R., Khandakar, Y.: Automatic time series forecasting: the forecast package for R. J. Stat. Softw. **27**(3), 1–22 (2008)
63. Ripley, B.: R: Seasonal decomposition of time series by loess. https://stat.ethz.ch/R-manual/R-devel/library/stats/html/stl.html
64. de Aguiar, P., Bourguignon, B., Khots, M., Massart, D., Phan-Than-Luu, R.: D-optimal designs. Chemometr. Intell. Lab. Syst. **30**(2), 199–210 (1995)
65. Ohno, S., Shiraki, T., Tariq, M.R., Nagahara, M.: Mean squared error analysis of quantizers with error feedback. IEEE Trans. Signal Process **65**(22), 5970–5981 (2017)
66. Zhang, L., Bi, J., Yuan, H.: Workload forecasting with hybrid stochastic configuration networks in clouds. In: 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS). pp. 112–116 (2018)
67. Zhou, Y., Wu, Y.: Analyses on influence of training data set to neural network supervised learning performance. In: Jin, D., Lin, S. (eds.) Adv. Comput. Sci. Intell. Syst. Environ., pp. 19–25. Springer, Berlin (2011)

68. Friedman, M.: A comparison of alternative tests of significance for the problem of *m* rankings. Ann. Math. Stat. **11**(1), 86–92 (1940)
69. Finner, H.: On a monotonicity problem in step-down multiple test procedures. J. Am. Stat. Assoc. **88**(423), 920–923 (1993)

Jitendra Kumar is an Assistant Professor in the Department of Computer Applications, National Institute of Technology Tiruchirappalli, India. He earned his doctorate from the National Institute of Technology Kurukshetra, India (An Institution of National Importance) in 2019. He has published several research articles in national and international journals and conferences of high repute. His current research interests include Cloud Computing, Machine Learning, Data Analytics, Parallel Processing.

Ashutosh Kumar Singh is working as a Professor and Head in the Department of Computer Applications, National Institute of Technology Kurukshetra, India. He has more than 18 years of research and teaching experience in various Universities of India, the UK, and Malaysia. He received his Ph.D. in Electronics Engineering from Indian Institute of Technology, BHU, India, and Post Doc from the Department of Computer Science, University of Bristol, UK. He is also a Charted Engineer from the UK. His research area includes Verification, Synthesis, Design, and Testing of Digital Circuits, Data Science, Cloud Computing, Machine Learning, Security, Big Data. He has published more than 160 research papers in different journals, conferences, and news magazines. He is the co-author of six books, which include 'Web Spam Detection Application using Neural Network', 'Digital Systems Fundamentals', and 'Computer System Organization & Architecture'. He has worked as an Editorial Board Member of International Journal of Networks and Mobile Technologies, International Journal of Digital Content Technology and its Applications. Also, he has shared his experience as a Guest Editor for the Pertanika Journal of Science and Technology. He is involved in reviewing the process of different journals and conferences such as; IEEE transaction of computer, IET, IEEE conference on ITC, ADCOM, etc.