JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 36, 981-992 (2020)
DOI: 10.6688/JISE.202009_36(5).0003

Adaptive Learning based Prediction Framework for Cloud Datacenter Networks' Workload Anticipation[*]

JITENDRA KUMAR[1,2] AND ASHUTOSH KUMAR SINGH[2]
[1]*Department of Computer Applications*
National Institute of Technology Tiruchirappalli
Tamilnadu, 620015 India
[2]*Department of Computer Applications*
National Institute of Technology Kurukshetra
Haryana, 136119 India
E-mail: jitendrakumar@ieee.org; ashutosh@nitkkr.ac.in

Cloud computing has effectively changed the computing industry by introducing the on-demand resources through virtualization. However, a cloud system suffers with several challenges including low resource utilization, high power consumption, security and many others. This paper introduces a neural network based workload forecasting model using differential evolution. The predictive framework is evaluated on five real world data traces. The forecast efficacy is compared with state-of-art approaches including back propagation and linear regression along with statistical analysis. It was observed that the proposed scheme reduced the forecast error up to 85.52% and 89.70% measured using RMSE and MAE respectively. The statistical analysis also validates the superiority of the proposed predictive framework as it received the best rank in the Friedman test analysis.

Keywords: workload forecasting, differential evolution, neural network, cloud computing, Google cluster trace

1. INTRODUCTION

Cloud computing has become one of the most popular word in the technology world. Today, it appears 58.6 million times on the Internet. According to MIT Technology Review, the term *cloud computing* was first coined by a group of Compaq Computers in 1996 [1, 2]. It delivers the computing resources as a service over Internet through a web application. A user can store his data and applications on a remote server which can be accessed at any time from any where across the globe with the help of an Internet connection. A cloud system must ensure to have on-demand self service, network access, resource pooling, elasticity, and measured services.

Elasticity is one of the most significant properties that expand or shrink the resources as the user demands increase or decrease over time. A cloud system ensures the elasticity using either of the two approaches called reactive and proactive scaling. In a reactive scaling method, the user has to wait for a certain amount of time before he gets the requested

Received July 29, 2019; accepted October 11, 2019.
Communicated by Jimson Mathew.
[*] This work is financially supported by Ministry of Electronics and Information Technology (MeitY), Government of India under Visvesvaraya PhD Scheme Fellowship.

resources as the approach adds or removes the resources only after the user demand is received. While a proactive approach estimates the user's demand and scale the resources in advance accordingly. Thus, a user does not need to wait to access the resources. But the estimations must be accurate otherwise the cloud system may suffer with the resource wastage, excess consumption of electricity and many other factors. Thus, the workload prediction plays a vital role in improving the quality of services (QoS) and quality of experience (QoE).

The workload prediction is being widely used for efficient resource management of cloud resources [3]. However, the high variability in the workloads (resource demands) introduces the complexity in workload estimation. In this paper, we introduce a neural network based predictive framework that exploits the differential evolution learning algorithm to optimize its synaptic connections. In addition, it also optimizes the length of patterns *i.e.* the number of historical workload instances to forecast next value.

Further, the paper is organized as follows: Section 2 briefs the recent key contribution in machine learning based forecasting schemes for cloud environment. Section 3 discusses the proposed learning based forecasting scheme followed by a discussion on experimental results in Section 4 along with a statistical analysis. Finally a conclusive remark is mentioned in Section 6.

2. RELATED WORK

Since, cloud server workloads can be indexed in time, the time series models can be used to anticipate the future workload [4]. Roy *et al*. [5] discussed the challenges of auto scaling in the cloud along with a forecasting approach using second order auto regressive moving average to achieve resource auto scaling in cloud environment. Tran *et al*. worked with seasonal ARIMA to show the efficacy of forecaster in long term forecasts [6] and produced the predictions ahead up to 168 hours. Calheiros *et al*. [7] and Ardagna *et al*. [8] used a forecasting methodology based on ARIMA and moving average (MA) respectively in the cloud environment for improved quality of services. In first approach, the authors analyzed the effect of forecasts and showed importance in accuracy to maintain the service level agreements along with user experience. On other hand, the second method proposed a distributed solution incorporating workload prediction and non linear optimization techniques to minimize the resource allocation cost along with a guaranteed SLA. The ARIMA model is also explored in improving the data center's network performance by achieving efficient network bandwidth allocation [9]. An adaptive workload prediction method is given by Liu *et al*. [4]. In this approach, the workload is classified into a number of classes and one of the prediction model is assigned to predict the expected load on the server. The authors used 0-1 integer programming for classifying the workload. In [10], the authors discovered the features and entities of auto scaling operations by analyzing the auto scaling techniques used by leading cloud service providers such as Amazon, Microsoft, and Google. The model permits a proactive analysis of workload patterns and estimation of the responsiveness of the auto scaling operations by modeling the extracted features and entities along with workload data. A comparative study of different time series forecasting model is carried out over eight real data traces [11]. Baldán *et al*. also studied and modeled the cloud data center workload forecasting problem as time series prediction [12]. The authors used a number of machine learning approaches along with ARIMA and ES to predict the workload instances. The cost reduction achieved by the proposed approach is 30% for both under and over provisioning cost. An experimental study over 21 prediction models is carried out in [13]. Authors modeled an application under four realistic workload patterns, two billing patterns, three types of predictive scaling. None of the model was observed to be universally best for each workload. Predictive scaling out with scaling in produced best cost efficiency and minimized the deadline miss rate of jobs. The different variants of ARMA class such as ARIMA, SARIMA (Seasonal ARIMA), ARFIMA (Fractionally Integrated ARIMA) with SSA (Singular Spectrum Analysis) are compared on CPU, Memory and bandwidth data traces obtained from the Wikimedia grid [14]. The performance of an ARIMA was found best on the network data trace while SSA outperformed other models over CPU and Memory data traces. Since none of the approach is universally good for all type of data traces, researchers suggested to use a combination of models. However, The time series forecasting models are unable to capture and model the presence of high non-linearity in cloud workloads as highlighted in [15].

A virtual machine workload prediction was proposed in [16] which uses the estimated information to determine whether an application is CPU intensive and/or memory intensive and resources are configured accordingly. An evolutionary neural network was used for workload prediction [17]. The approach implements particle swarm optimization, differential evolution, and covariance matrix adaptation evolutionary strategy learning algorithms and compares their performance. The concept of sliding window has been used with neural network and linear regression for estimating the future workloads [18]. The execution time of tasks is anticipated to achieve better scheduling of tasks [19]. The approach also extends one of the existing scheduling algorithms to enable the usage of anticipated task costs for online partitioning and scheduling. Duy *et al*. [20] also explored the neural network to improve the host load forecast accuracy. A dynamic environment is used to illustrate the feasibility of the approach. Garg *et al*. [21] introduced a model to address the issues of resource allocation in a datacenter that hosts different categories of workloads. The predictive framework estimates the resource utilization of non-interactive and transactional applications and schedules the workloads to achieve better usage of resources. In addition, the profit of service providers and QoS requirements are maintained. A number of neural network based prediction approaches have been proposed to improve the forecast accuracy including [22-29]. Unlike any linear model, the neural network based predictive framework are capable of modeling the nonlinearity in the workload traces.

3. PREDICTIVE FRAMEWORK

The predictive framework is responsible to anticipate the future workload, where workload can be defined as the number of HTTP requests on web server, computing resource demands and others. The workflow of proposed predictive framework is shown in Fig. 1. First, the model extracts and aggregates the workload information that is analyzed to optimize the length of learning window. The workload information is preprocessed and organized as per the length of learning window to fed into model learning module. Then the trained model's forecast accuracy is evaluated on test data. The ratio of training and testing data is 60:40.

3.1 Pattern Length Optimization

The number of workload instances being analyzed highly affects the quality of forecasts. We define these instances as learning window of length ℓ or pattern length. The proposed model develops the process of learning the optimal length of learning window. It analyzes the workload trace and computes the autocorrelation for k time lags using Eq. (1), where $k = \{1, 2, \ldots, 40\}$ and T denotes the length of workload trace. The number of lags with significant autocorrelation (s_l) is identified using Eq. (2), where τ_ρ denotes the threshold value of significant auto correlation. The process computes the autocorrelation for $\lambda = 40$ time lags, where value of λ is selected based on experimental analysis. If each instance is having significant autocorrelation, the workload trace is differentiated using Eq. (3), where $\tilde{y}$ denotes the differenced workload. The same process is repeated again to identify the optimal length. In case of differenced trace, if each time lag is having significant autocorrelation, the λ is assigned to be the length of pattern.

$$\rho_k = \frac{\sum_{t=1}^{T-k}(Y_t - \bar{Y})(Y_{t+k} - \bar{Y})}{\sum_{t=1}^{T}(Y_t - \bar{Y})^2} \tag{1}$$

$$s_l = \begin{cases} k; & \text{if } \rho_k \geq \tau_\rho \\ 0; & \text{otherwise} \end{cases} \qquad \forall k \tag{2}$$

$$\tilde{y}_t = y_{t+1} - y_t \tag{3}$$

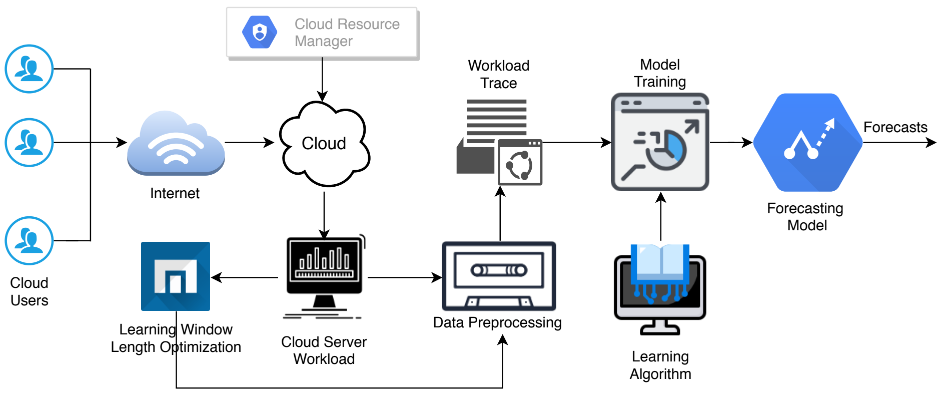


Fig. 1. Predictive framework model.

3.2 Network Optimization

The predictive framework is developed using a layered structure of an artificial neural network that contains $\iota - \bar{h} - \kappa$ neurons in input, hidden and output layers respectively. The network learns from the training data and optimizes its synaptic connection weights using an optimization algorithm. The proposed framework considers the differential evolution due to its simple and powerful optimization approach [30]. It is a population based algorithm that uses vector difference to search for an optimal solution in the problem space. In order to optimize the connection weights of proposed scheme, it first randomly initializes P vectors or solutions using Eq. (4), where $lb_j = -1$ and $ub_j = 1$ are the lower and upper bounds respectively while *rand* is random number in the range [0, 1]. The length of network is decided to be $D = ((\imath + 1) \times \bar{h} + \bar{h} \times 1) \Rightarrow \imath(\bar{h} + 2)$ as it is fully connected feed forward network.

Algorithm 1 : Cloud resource demand predictive model pseudocode.

```
Input: $Y, L, P, G, T_\rho, d_o, PWS$
Output: $\hat{Y}$
 1: Identify the length of pattern ($\ell$) using autocorrelation analysis
 2: $\imath = \ell, \bar{h} = \lfloor 2\ell/3 \rfloor, \kappa = 1$
 3: Initialize P random networks of length $D$ (number of synaptic connections in the network)
 4: Compute the fitness of each network by evaluating them on training data
 5: while termination criteria is not met do
 6:   for each generation $G$ do
 7:     for each solution $i$ do
 8:       Generate $r_1 \neq r_2 \neq i \in (1, P)$ and $cp_{rand} \in (1, P)$
 9:       Apply DE/rand/1 mutation strategy and single point crossover
10:     end for
11:     Compute the fitness of each offspring network by evaluating them on training data
12:     Select P better individuals from offspring and current population for next iteration
13:   end for
14: end while
```

$$\eta_{i,j} = lb_j + rand \times (ub_j - lb_j) \tag{4}$$

The performance of each network is evaluated on training data and a fitness value is computed using an objective function such as minimization of forecast error. Further, a number of solutions are generated by a process of recombination using mutation and crossover operators. The mutation operation prevents the premature convergence by escaping a solution from local optima. On the other hand, crossover recombines the existing solutions to explore the search space to find an optimal solution. The approach uses *DE/rand/1* mutation operator that uses the random selection of members and one differenced vector to perturb the networks as shown in Eq. (5). The r_1, r_2 are the distinct random numbers in (1, D) range and α is mutation rate in [0, 1) range. The k symbolizes the iteration or generation number.

$$\mu_{i,j}^{k+1} = \eta_{i,j}^{k} + \alpha \times (\eta_{r_1,j}^{k} - \eta_{r_2,j}^{k}) \qquad r_1 \neq r_2 \neq i \tag{5}$$

The mutants further mate with the solutions from population to generate offspring solutions (υ). The predictive scheme uses the single point crossover that selects η_i and μ_i for operation. Then a random number (cp_{rand}) in (1, D) range is generated that splits the solutions into two parts and tails of both solutions are exchanged to generate the offspring networks. The offspring solutions are evaluated on the training data and better offsprings replace the respective networks from the population. The selection of the population for next iteration is carried out using the survival of the fitter between η_i and υ_i as shown in Eq. (6).

$$\eta_i^{k+1} = \begin{cases} \upsilon_i^k & \text{if } f(\upsilon_i^k) < f(\eta_i^k) \\ \eta_i^k & \text{otherwise} \end{cases} \tag{6}$$

Further, we compute the computational time complexity of the proposed scheme step y step. The initialization of P networks (line 3) consumes $\mathcal{O}(\mathrm{P} \times D)$. Since $D = \bar{h}(\iota + 2)$ and $\bar{h} < \iota$, it becomes $\mathcal{O}(\mathrm{P} \times \bar{h} \times (\iota + 2)) \Rightarrow \mathcal{O}(\iota \times \bar{h} \times \mathrm{P}) \Rightarrow \mathcal{O}(\iota^2 \times \mathrm{P})$. The fitness evaluation of the population (line 4) needs $\mathcal{O}(m \times \iota^2 \times \mathrm{P})$. The generation of random numbers (line 8) takes $\mathcal{O}(\mathrm{P})$ as it generates three random numbers P times. The generation of offspring solutions involve the mutation and crossover operations that consumes $\mathcal{O}(\iota)$ to produce one offspring and $\mathcal{O}(\iota^2 \times \mathrm{P})$ to generate P networks. Since each offspring needs to be evaluated (line 11) for further processing, it takes $\mathcal{O}(m \times \iota^2 \times \mathrm{P})$. The next step *i.e.* selection (line 12) requires P comparisons that needs $\mathcal{O}(\mathrm{P})$ time complexity. The total complexity of one iteration becomes $\mathcal{O}(m \times \iota^2 \times \mathrm{P})$. For a maximum iteration (G), it becomes $\mathcal{O}(m \times \iota^2 \times \mathrm{P} \times G)$.

4. PREDICTION RESULTS & ANALYSIS

This section evaluates the forecast accuracy of the proposed scheme and compares it with state-of-art learning based forecast methods. The forecasts are carried out on different prediction window size (PWS) that defines the time interval between two consecutive forecasts. We evaluated the forecast accuracy using root mean squared error (RMSE) (7) and mean absolute error (MAE) (8). A machine equipped with two Intel® Xeon® E5-2630 v4 processors. The machine contains main memory of 128 GB. We implemented the predictive approach using MATLAB 2017a. Five different data traces are used to evaluate the forecast accuracy including three web server traces (D_1, D_2, and D_3) and two traces (D_4 and D_5) of Google cluster trace [31, 32].

$$\mathrm{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(w_i - \hat{w}_i)^2} \tag{7}$$

$$\mathrm{MAE} = \frac{1}{m}\sum_{i=1}^{m}|w_i - \hat{w}_i| \tag{8}$$

4.1 Short Term Forecast Evaluation

The forecasts are evaluated on short term intervals *i.e.* up to 30 minutes. Tables 1 and 2 list the forecast accuracy on 1, 5, 10, 20, and 30 minutes interval using RMSE and MAE metrics. It can be observed that the forecast for 1 minute interval received better quality. It was also noticed that the forecast quality drops down as the forecast interval increases up to certain value and improves again. The behavior was observed due to fact that as the forecast interval increases beyond certain value, the quality of patterns improves and can be easily captured as it was not the case with lower values of prediction intervals.

Table 1. Short term forecast evaluation using RMSE of proposed scheme.

PWS (min)	D_1	D_2	D_3	D_4	D_5
1	3.14E-1	1.19E-1	2.41E-1	2.82E+2	3.53E+2
5	7.15E-2	2.20E-1	4.39E-1	1.66E+2	1.84E+2
10	5.98E-2	2.81E-1	2.37E-1	1.62E+3	1.29E+3
20	1.28E-1	1.87E-1	2.63E-1	3.08E+3	2.45E+3
30	6.16E-2	1.33E-1	3.97E-1	8.13E+2	9.91E+2

Table 2. Short term forecast evaluation using MAE of proposed scheme.

PWS (min)	D_1	D_2	D_3	D_4	D_5
1	2.13E-1	8.87E-2	1.82E-1	1.69E+2	1.82E+2
5	5.03E-2	1.85E-1	4.08E-1	1.07E+2	9.79E+1
10	4.60E-2	2.31E-1	2.07E-1	9.34E+2	7.37E+2
20	9.42E-2	1.50E-1	2.29E-1	1.87E+3	1.48E+3
30	4.34E-2	1.09E-1	3.36E-1	5.75E+2	6.40E+2

4.2 Long Term Forecast Evaluation

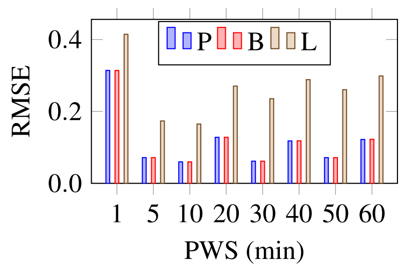
Similarly, a study on long term forecast *i.e.* beyond 30 minutes is carried out. Tables 3 and 4 list the forecast accuracy on 40, 50, and 60 minutes interval using RMSE and MAE metrics. Based on the forecast accuracy, it is clear that no trend is detected. However, in most of the cases the forecast accuracy drops down for 60 minute prediction interval that may be due to availability of less amount of training samples.

Table 3. Long term forecast evaluation of proposed scheme based on RMSE.

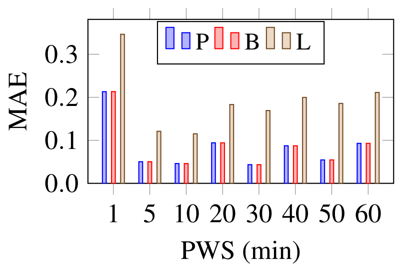
PWS (min)	D_1	D_2	D_3	D_4	D_5
40	1.18E-1	2.29E-1	5.35E-1	5.73E+3	4.64E+3
50	7.15E-2	2.86E-1	4.83E-1	7.03E+3	5.62E+3
60	1.22E-1	2.43E-1	5.83E-1	5.82E+3	5.25E+3

Table 4. Long term forecast evaluation of proposed scheme based on MAE.

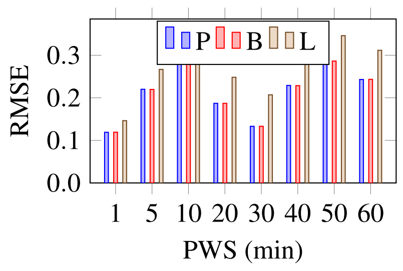
PWS (min)	D_1	D_2	D_3	D_4	D_5
40	8.73E-2	1.96E-1	4.48E-1	3.71E+3	2.93E+3
50	5.44E-2	2.34E-1	4.05E-1	4.63E+3	3.65E+3
60	9.29E-2	2.06E-1	4.99E-1	5.36E+3	4.66E+3



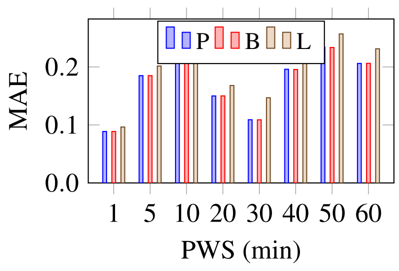
(a) NASA Trace (RMSE)



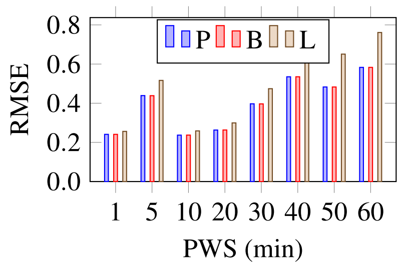
(b) NASA Trace (MAE)



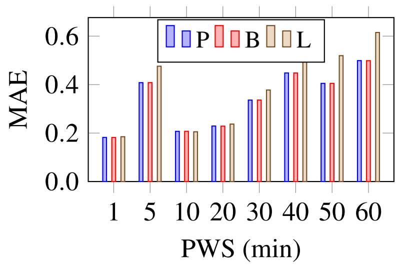
(c) Calgary Trace (RMSE)



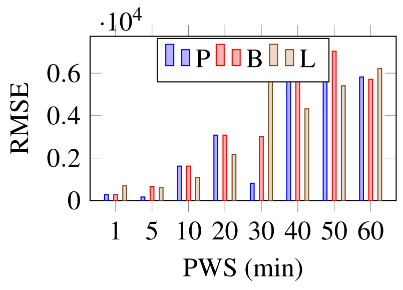
(d) Calgary Trace (MAE)



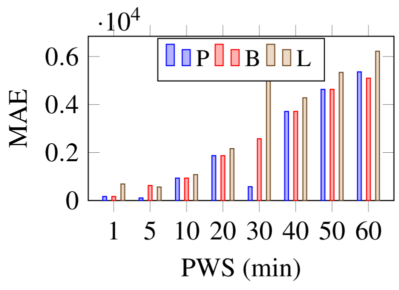
(e) Saskatchewan Trace (RMSE)



(f) Saskatchewan Trace (MAE)

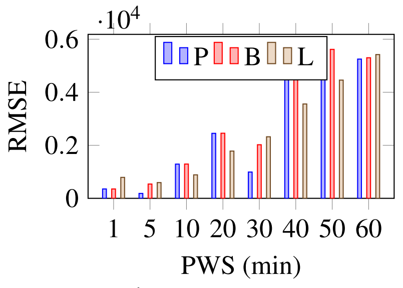


(g) CPU Trace (RMSE)

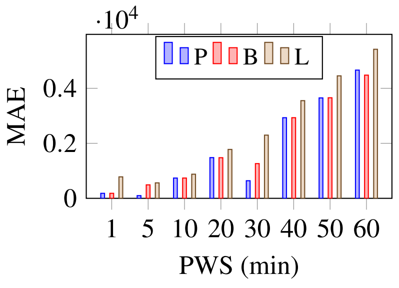


(h) CPU Trace (MAE)

Fig. 2. Accuracy comparison using five different data traces.



(i) Memory Trace (RMSE)



(j) Memory Trace (MAE)

Fig. 2. (Cont'd) Accuracy comparison using five different data traces.

5. COMPARATIVE & STATISTICAL ANALYSIS

This section compares the performance of the proposed scheme with two state-of-art learning algorithms based prediction schemes. The proposed predictive framework is trained using back propagation and linear regression learning algorithms and performance of each model is measured and compared as shown in Fig. 3, where P, B, and L represents the proposed scheme, back propagation, and linear regression based predictive models respectively. It can be observed that the proposed model trained using differential evolution received better forecast accuracy in most of cases. For instance, the proposed model reduces the forecast error (RMSE) up to 75.16% and 65.85% for CPU and memory data traces over back propagation based forecasting method. Similarly, the forecast RMSE over linear regression based method is reduced up to 35.76%, 16.28%, 85.52%, 69.00%, and 74.34% for all five data traces respectively.

In order to observe the performance of each approach, we conducted the statistical analysis using Friedman and Finner post-hoc method. Friedman test considers a null hypothesis (H_0) that assumes the equivalence of mean of the results for all approaches. It was observed that the test rejects the H_0 for RMSE and MAE based experiments. Table 5 lists the mean rank for both metrics and we observed that differential evolution based predictive framework attained the lowest ranks that indicates the best performance among considered algorithms. We also conducted a post-hoc test using Finner method that assumes that the mean of the performance of both algorithms in each pair is same. Table 6 shows that the H_0 is rejected when the performance of the proposed and linear regression (LR) is compared whereas the test accepts the H_0 on comparing the performance of proposed and back propagation (BP) based prediction models.

Table 5. Friedman test mean ranks.

Algorithm	Rank (RMSE)	Rank (MAE)
Proposed	1.625	1.425
Back propagation	1.800	1.650
Linear Regression	2.575	2.925

Table 6. Post-hoc analysis using Finner test.

	RMSE			MAE		
	Statistics	Adjusted p-value	Result	Statistics	Adjusted p-value	Result
Proposed vs LR	4.24853	0.00004	$H_0.R$	6.70820	0.00000	$H_0.R$
Proposed vs BP	0.78262	0.43385	$H_0.A$	1.00623	0.31430	$H_0.A$

6. CONCLUDING REMARKS

Workload prediction has been effectively used to improve the cloud resource management. In this paper we presented a resource demand forecast mechanism that learns the network weights using differential evolution based learning scheme. The proposed predictive scheme is evaluated on five different workload traces. The prediction results validate the better forecasts achieved by the proposed scheme as it reduces the error from forecasts. The convincing experimental results show that the RMSE and MAE are reduced up to 85.52% and 89.70% respectively. The predictive approach can be further extended to achieve self-parameter learning and multi step forecasts.

REFERENCES

1. A. Regalado, "Who coined 'Cloud Computing'?" *MIT Technology Review*, Technical Report, 2011.
2. "Internet solutions division strategy for cloud computing," Compaq Computer Corporation, Technical Report, 1996.
3. S. S. Gill and R. Buyya, "A taxonomy and future directions for sustainable cloud computing: 360 degree view," *ACM Computing Surveys*, Vol. 51, 2018, pp. 104:1-104:33.
4. C. Liu, C. Liu, Y. Shang, S. Chen, B. Cheng, and J. Chen, "An adaptive prediction approach based on workload pattern discrimination in the cloud," *Journal of Network and Computer Applications*, Vol. 80, 2017, pp. 35-44.
5. N. Roy, A. Dubey, and A. Gokhale, "Efficient autoscaling in the cloud using predictive models for workload forecasting," in *Proceedings of IEEE 4th International Conference on Cloud Computing*, 2011, pp. 500-507.
6. V. G. Tran, V. Debusschere, and S. Bacha, "Hourly server workload forecasting up to 168 hours ahead using seasonal arima model," in *Proceedings of IEEE International Conference on Industrial Technology*, 2012, pp. 1127-1131.
7. R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, "Workload prediction using ARIMA model and its impact on cloud applications' QoS," *IEEE Transactions on Cloud Computing*, Vol. 3, 2015, pp. 449-458.
8. D. Ardagna, S. Casolari, M. Colajanni, and B. Panicucci, "Dual time-scale distributed capacity allocation and load redirect algorithms for cloud systems," *Journal of Parallel and Distributed Computing*, Vol. 72, 2012, pp. 796-808.
9. J. Cao, X. Zhu, F. Dong, B. Liu, Z. Ma, and H. Min, "Time series based bandwidth allocation strategy in cloud datacenter," in *Proceedings of International Conference on Advanced Cloud and Big Data*, 2016, pp. 228-233.

10. M. N. A. H. Khan, Y. Liu, H. Alipour, and S. Singh, "Modeling the autoscaling operations in cloud with time series data," in *Proceedings of IEEE 34th Symposium on Reliable Distributed Systems Workshop*, 2015, pp. 7-12.
11. A. Adegboyega, "Time-series models for cloud workload prediction: A comparison," in *Proceedings of IFIP/IEEE Symposium on Integrated Network and Service Management*, 2017, pp. 298-307.
12. F. J. Baldán, S. Ramírez-Gallego, C. Bergmeir, F. Herrera, and J. M. Benítez, "A forecasting methodology for workload forecasting in cloud systems," *IEEE Transactions on Cloud Computing*, Vol. 6, 2018, pp. 929-941.
13. I. K. Kim, W. Wang, Y. Qi, and M. Humphrey, "Empirical evaluation of workload forecasting techniques for predictive cloud resource scaling," in *Proceedings of IEEE 9th International Conference on Cloud Computing*, 2016, pp. 1-10.
14. A. S. Kumar and S. Mazumdar, "Forecasting hpc workload using arma models and ssa," in *Proceedings of International Conference on Information Technology*, 2016, pp. 294-297.
15. M. Amiri and L. Mohammad-Khanli, "Survey on prediction models of applications for resources provisioning in cloud," *Journal of Network and Computer Applications*, Vol. 82, 2017, pp. 93-113.
16. S. Kumaraswamy and M. K. Nair, "Intelligent VMs prediction in cloud computing environment," in *Proceedings of International Conference on Smart Technologies for Smart Nation*, 2017, pp. 288-294.
17. K. Mason, M. Duggan, E. Barrett, J. Duggan, and E. Howley, "Predicting host CPU utilization in the cloud using evolutionary neural networks," *Future Generation Computer Systems*, Vol. 86, 2018, pp. 162-173.
18. S. Islam, J. Keung, K. Lee, and A. Liu, "Empirical prediction models for adaptive resource provisioning in the cloud," *Future Generation Computer Systems*, Vol. 28, 2012, pp. 155-162.
19. J. Li, X. Ma, K. Singh, M. Schulz, B. R. de Supinski, and S. A. McKee, "Machine learning based online performance prediction for runtime parallelization and task scheduling," in *Proceedings of IEEE International Symposium on Performance Analysis of Systems and Software*, 2009, pp. 89-100.
20. T. V. T. Duy, Y. Sato, and Y. Inoguchi, "Improving accuracy of host load predictions on computational grids by artificial neural networks," *International Journal of Parallel, Emergent and Distributed Systems*, Vol. 26, 2011, pp. 275-290.
21. S. K. Garg, A. N. Toosi, S. K. Gopalaiyengar, and R. Buyya, "Sla-based virtual machine management for heterogeneous workloads in a cloud datacenter," *Journal of Network and Computer Applications*, Vol. 45, 2014, pp. 108-120.
22. J. J. Prevost, K. Nagothu, B. Kelley, and M. Jamshidi, "Prediction of cloud data center networks loads using stochastic and neural models," in *Proceedings of the 6th International Conference on System of Systems Engineering*, 2011, pp. 276-281.
23. Y.-C. Chang, R.-S. Chang, and F.-W. Chuang, "A predictive method for workload forecasting in the cloud environment," in *Proceedings of International Conference on Advanced Technologies, Embedded and Multimedia for Human-centric Computing*, 2014, pp. 577-585.

24. J. Kumar and A. K. Singh, "Workload prediction in cloud using artificial neural network and adaptive differential evolution," *Future Generation Computer Systems*, Vol. 81, 2018, pp. 41-52.
25. Y. Lu, J. Panneerselvam, L. Liu, and Y. Wu, "Rvlbpnn: A workload forecasting model for smart cloud computing," *Scientific Programming*, Vol. 2016, 2016, pp. 1-9.
26. G. Kousiouris, A. Menychtas, D. Kyriazis, S. Gogouvitis, and T. Varvarigou, "Dynamic, behavioral-based estimation of resource provisioning based on high-level application terms in cloud platforms," *Future Generation Computer Systems*, Vol. 32, 2014, pp. 27-40.
27. J. Kumar and A. K. Singh, "Cloud resource demand prediction using differential evolution based learning," in *Proceedings of the 7th International Conference on Smart Computing and Communications*, 2019, pp. 1-5.
28. J. Kumar and A. K. Singh, "Dynamic resource scaling in cloud using neural network and black hole algorithm," in *Proceedings of the 5th International Conference on Eco-friendly Computing and Communication Systems*, 2016, pp. 63-67.
29. J. Kumar, R. Goomer, and A. K. Singh, "Long short term memory recurrent neural network (LSTM-RNN) based workload forecasting model for cloud datacenters," *Procedia Computer Science*, Vol. 125, 2018, pp. 676-682.
30. K. V. Price, "Differential evolution: a fast and simple numerical optimizer," in *Proceedings of Biennal Conference of the North American Fuzzy Information Processing Society*, 1996, pp. 524-527.
31. "Traces available in the internet traffic archive," ftp://ita.ee.lbl.gov/html/, 2019.
32. C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *Proceedings of ACM Symposium on Cloud Computing*, 2012, pp. 1-18.

Jitendra Kumar is an Assistant Professor in the Department of Computer Applications, National Institute of Technology Tiruchirappalli, Tamilnadu, India. He conducts research in predictive modeling, nature-inspired algorithms, cloud computing. He has published more than 25 articles in various journals and conferences of high repute. He received his Ph.D. from National Institute of Technology Kurukshetra, India.

Ashutosh Kumar Singh is Professor in the Department of Computer Applications, National Institute of Technology Kurukshetra, India. He conducts research in artificial intelligence, cyber physical systems, data science, quantum computing, digital circuit logic. He has a vast experience of more than 20 years of teaching and administration across globe. He has published more than 200 research articles in various journals and conferences of high repute. He received his PhD from Indian Institute of Technology-BHU Varanasi, India.