Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 125 (2018) 676–682

Procedia Computer Science

www.elsevier.com/locate/procedia

6th International Conference on Smart Computing and Communications, ICSCC 2017, 7-8 December 2017, Kurukshetra, India

Long Short Term Memory Recurrent Neural Network (LSTM-RNN) Based Workload Forecasting Model For Cloud Datacenters

Jitendra Kumar[a,*], Rimsha Goomer[b], Ashutosh Kumar Singh[a]

[a]*Department of Computer Applications, National Institute of Technology, Kurukshetra, India*
[b]*Department of Computer Science, Viterbi School of Engineering, University of Southern California, USA*

Abstract

In spite of various gains, cloud computing has got few challenges and issues including dynamic resource scaling and power consumption. Such affairs cause a cloud system to be fragile and expensive. In this paper we address both issues in cloud datacenter through workload prediction. The workload prediction model is developed using long short term memory (LSTM) networks. The proposed model is tested on three benchmark datasets of web server logs. The empirical results show that the proposed method achieved high accuracy in predictions by reducing the mean squared error up to 3.17×10^{-3}.

Peer-review under responsibility of the scientific committee of the 6th International Conference on Smart Computing and Communications

Keywords: Cloud Computing, Resource Scaling, Forecasting, Deep Learning

1. Introduction

Cloud computing has turned into a revolutionary milestone of computing world. It has completely transformed the means of computing. In 2015, Amazon Web Services (AWS) generated $7.88B in revenue with Q4 2015, up 69% over last year [1]. The technology is widely being adapted by individuals, organizations, governments, academia. Every individual uses the technology in one way or other. The daily life examples include social networking, e-governance, online shopping, health care and others [2–6]. In 2016, spending on public cloud Infrastructure as a service hardware and software is forecast to reach $38B, growing to $173B in 2026 [1]. It is estimated that, world will have 50 billion connected devices by 2020. Interconnection among these sensors would cause generation of data in massive amount. This data must be stored and precessed to get benefits from it. The traditional infrastructure are inadequate to store and process such amount of data in an efficient manner. The cloud architectures have emerged as one possible solution for the same.

*Corresponding author

E-mail addresses: jitendrakumar@ieee.org (Jitendra Kumar)., goomer@usc.edu (Rimsha Goomer)., ashutosh@nitkkr.ac.in (Ashutosh Kumar Singh).

1877-0509
Peer-review under responsibility of the scientific committee of the 6th International Conference on Smart Computing and Communications
10.1016/j.procs.2017.12.087

Cloud technology is equipped with a number of qualities such as flexibility, disaster recovery, mobility, storage, on demand resources and several others. Despite of such aids the technology has some points of concern including efficient resource scaling, power consumption, virtual machine placement, security, privacy, resource utilization etc. This paper deals with two important factors i.e. dynamic resource scaling and power consumption by the means of workload forecasting. The dynamic resource scaling enables a cloud system to retain quality of service (QoS) as per the service level agreements (SLAs). On the other hand reduction in power consumption helps in developing an eco-friendly and economic cloud. If we can determine server's accurate future workload well ahead in time, the resources can be scaled up or down accordingly. For instance, if the estimated workload is more than current workload, the required resources can be added before actual load arrives to keep QoS up. On other hand if expected workload is less than the present load, the resources can be shut down to shorten power consumption.

The problem of workload prediction in cloud datacenter has been addressed using several different approaches. These approaches can be classified in two main categories, time series models and machine learning techniques based models. First class of models such as auto regression (AR), moving average (MA), exponential smoothing (ES), auto regressive integrated moving average (ARIMA) and others have been widely used in forecasting the load [7–12]. But unfortunately these methods could not prove themselves in long time prediction. However people have come up with machine learning techniques to cope the issues of these methods. Recently machine learning approaches such as self organizing feature map (SOFM), support vector machines (SVMs), *k*NN, neural networks, and nature inspired algorithms are extensively adopted to predict the upcoming workload on the server. L. Cao proposed a self organizing map (SOM) and support vector machines (SVMs) based predictive framework for time series data [13]. The approach employed two phases i.e. data is clustered using SOM and forecasting is performed using SVM. Bat et al. [14] suggested a two tier architecture using *k* Nearest Neighbors (*k*NN) for financial time series prediction. Neural network was used in [15] to model workload variation in multimedia designs. We also advocated the use of neural network in workload estimation model along with nature inspired optimization and adaptive differential evolution in [16] and [17] respectively. This paper presents a workload prediction method using Long Short Term Memory (LSTM) Recurrent Neural Network (RNN).

Rest of the paper is organized as section 2 briefs the basics of workload prediction and LSTM networks. Section 3 provides a discussion on proposed model followed by analysis of obtained results in section 4. Paper concludes with conclusive note and a word on future directions in section 5.

2. Preliminaries

2.1. Workload Prediction

It is a course of forecasting the expected variations in system's future workload. This enables one to answer questions such as what amount of emails will arrive in near future. Since past plays a critical role in predicting the future, it becomes necessary to analyze the history of datacenter. In this process model tries to extract a pattern from the historical data. The extracted pattern of workload is used to forecast the upcoming workload. In our work we have used n continuous previous samples in predicting the load considering the fact that most recent event affects most comparatively to other previous events. Mathematically a linear prediction model can be represented as given in (1). Where x_i is the actual workload on the server at time instance i and a_i defines the importance of x_i in predicting $\hat{y}_{t+1}$.

$$\hat{y}_{t+1} = a_1 \times x_1 + a_2 \times x_2 + \cdots + a_n \times x_n \tag{1}$$

2.2. LSTM Network

Deep learning or deep structured learning can be defined as special kind of neural networks composed of multiple layers. These networks are better than traditional neural network in persisting the information from previous event. Recurrent neural network (RNN) is one such machine that has a combination of networks in loop. The networks in loop allow the information to persist. Each network in the loop takes input and information from previous network,

performs the specified operation and produces output along with passing the information to next network. Some applications require only recent information while others may ask for more from past. The common recurrent neural networks lag in learning as the gap between required previous information and the point of requirement increases to a large extent. But fortunately Long Short Term Memory (LSTM) Networks [18], a special form of RNN are capable in learning such scenarios. These networks are precisely designed to escape the long term dependency issue of recurrent networks.

LSTMs are good in remembering information for long time. Since more previous information may affect the accuracy of model, LSTMs become a natural choice of use. Typical LSTM module called repeating module has four neural network layers interacting in a unique fashion as shown in Fig. 1. Module has three gate activation functions σ_1, σ_2, and σ_3 and two output activation functions ϕ_1 and ϕ_2 as depicted in Fig. 1. The symbol π and Σ represent element wise multiplication and addition respectively. The concatenation operation is represented by symbol ($\bullet$) bullet. The fundamental component of LSTMs is cell state, a line running from *Memory from Previous Block* (S_{t-1}) to *Memory from Current Block* (S_t). It allows the information to flow straight down the line. Network can decide the amount of previous information to flow. It is controlled through first layer (σ_1). The operation performed by this layer is given in (2).

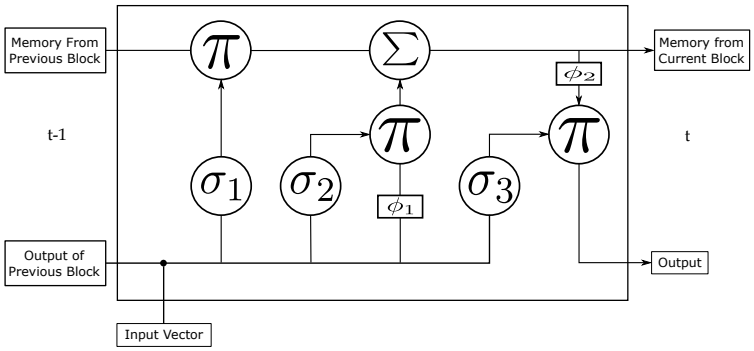


Figure 1: Repeating Module of LSTM

$$cf_t = \sigma_1(W_{cf} \cdot [O_{t-1}, x_t] + b_{cf}) \tag{2}$$

$$I_t = \sigma_2(W_I \cdot [O_{t-1}, x_t] + b_I) \tag{3}$$

$$\tilde{S}_t = tanh(W_S \cdot [O_{t-1}, x_t] + b_S) \tag{4}$$

$$S_t = cf_t \times S_{t_1} + I_t \times \tilde{S}_{t-1} \tag{5}$$

The new information to be stored in cell state is computed using two network layers. A sigmoid layer (σ_2) that decides values to update (I_t) (see (3)) and tanh layer ϕ_1 that evolves a vector of new candidate values ($\tilde{S}_t$) as shown in (4). The combination of both to be added in the state. Finally cell state is updated using (5).

3. Prediction Using LSTM-RNN

The future workload information is one of the essential parameters in dynamic resource scaling. The efficient resource scaling leads a system to be cost effective. A good resource scaling method also helps in reducing the power consumption by shutting off unused resources. Thus the system becomes eco-friendly too.

In the proposed model output of predictive unit is fed into a device called *resource manager* that also takes the current state of datacenter into account before taking resource scaling decisions as shown in Fig. 2. If available

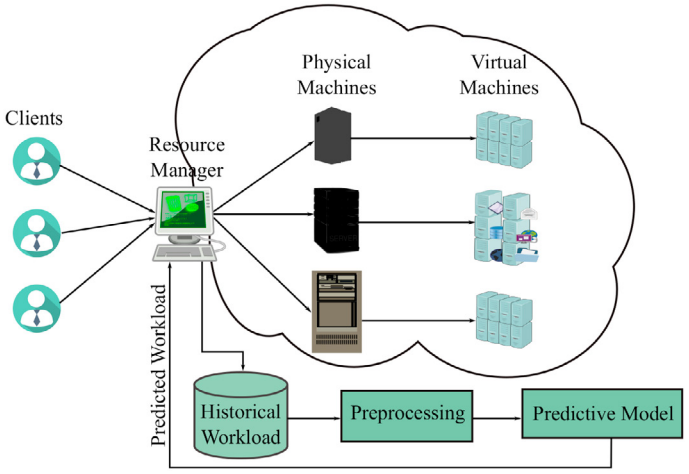


Figure 2: Cloud Datacenter workload forecasting model

1: Set *ip_units, lstm_units, op_units* and *optimizer* to define LSTM Network (L)
2: Normalize the dataset (D_i) into values from 0 to 1 using (6)
3: Select *training window size (tw)* and organize D_i accordingly
4: **for** *n_epochs* and *batch_size* **do**
5: Train the Network (L)
6: **end for**
7: Run Predictions using L
8: Calculate the *loss_function* using (7)

Figure 3: Pseudocode for Workload Prediction using LSTM

resources are not enough to fulfill the expected load, the resources can be scaled up. While in the case where resources are more than required, these can be scaled down.

In preprocessing step, data is normalized in the range (0,1) using (6) and reshaped according to the training window size (*tw*). We define training window to be a set of patterns used to predict next pattern. For example x_1, x_2, x_3, x_4, x_5 are used to predict value for x_6 then training window size is 5. The model (L) is composed with 4 units of repeating module R_1, R_2, R_3, and R_4. The first module (R_1) takes input vector (x_1, x_2, $\cdots$, x_{tw}) of length training window (tw) along with previous information. Initially this information is assigned to be 0. The second module (R_2) takes input (x_2, x_3, $\cdots$, x_{tw+1}) in combination with output of first module. Similarly third and forth modules (R_3 and R_4) take input vectors along with output of adjacent previous unit. Finally the fourth unit (R_4) produces predicted value y_{tw+4}. To measure the accuracy of model we computed the mean squared error. Objective of the network is to minimize the mean square error as given in (7). The algorithmic representation of proposed approach is given in Fig. 3.

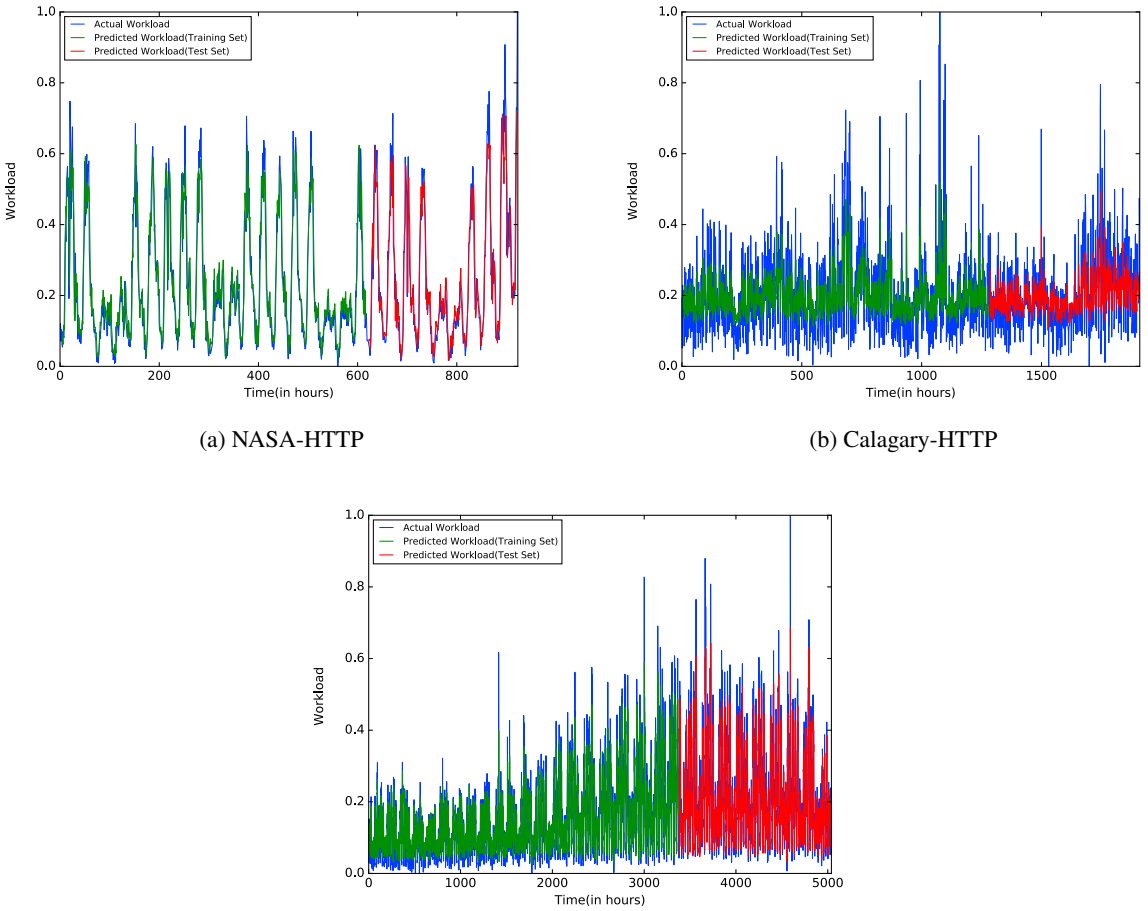
$$X_{norm} = \frac{X - min(X)}{max(X) - min(X)} \tag{6}$$

$$f_{obj} = min(\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2) \tag{7}$$

Where $\hat{y}_i$ and y_i are the predicted and actual values of workload at time instance i respectively. And N is the number of data samples.

4. Experimental Results

The experiments were performed on Intel® Core™ I5-M520 processor of 2.40GHz clock speed having 4 GB of memory. We opted Python along with Keras library as a tool for implementation and simulations. We performed the experiments on three datasets; HTTP traces of NASA server, Calgary server, and Saskatchewan server collected and analyzed by [19]. In this paper we referred these datasets as D_1, D_2 and D_3 respectively. NASA-HTTP (D_1) contains two traces having two months worth of all HTTP requests to the NASA Kennedy Space Center WWW server in Florida. Calgary-HTTP (D_2) contains approximately one year's worth of all HTTP requests to the University of Calgary's Department of Computer Science WWW server located at Calgary, Alberta, Canada. Saskatchewan-HTTP (D_3) data is seven months of HTTP logs from a University WWW server. The traces are stored in ASCII files with one line per request. The attributes of data are *host*, *timestamp*, *request*, *HTTP reply code* and *bytes in the reply*.



(a) NASA-HTTP

(b) Calagary-HTTP

(c) Saskatchewan-HTTP

Figure 4: Actual Load vs. Predicted Load (PWS = 60 Minutes)

The detailed study was carried out on the workload forecasting with 1, 5, 10, 20, 30, and 60 minutes of prediction window size (PWS). PWS defines the duration between two consecutive forecasts. Training of the model is carried out with *tw* of size 20. The data was divided in two parts called training and test data in the ratio of 60:40. Fig. [4a-4c] depict the predictions carried out with 60 minute interval for D_1, D_2 and D_3 respectively. From the results it is clearly evident that the predicted load for D_1 and D_3 is very close to actual load in most of the time instances. The model is

unable to produce the predictions for D_2 with desired accuracy due to presence of high variability in the workload of Calgary server.

Table 1: Prediction accuracy comparison

PWS	Mean Squared Error [a]								
	LSTM-RNN			Blackhole [16]			Back Propagation [20]		
	D_1	D_2	D_3	D_1	D_2	D_3	D_1	D_2	D_3
1	13.06	3.42	5.00	21.45	6.03	1.61	243.17	297.31	40.23
5	4.79	4.10	3.17	8.45	5.99	2.51	302.25	290.01	336.95
10	6.66	6.11	5.26	12.03	14.78	5.55	281.80	286.55	290.40
20	7.01	5.99	5.56	7.78	12.50	8.82	338.06	278.27	318.29
30	6.43	7.12	4.79	9.06	17.77	8.03	278.42	500.08	507.30
60	5.59	8.03	4.50	23.13	19.70	9.30	333.85	297.02	286.33

[a] In order of 10^{-3}

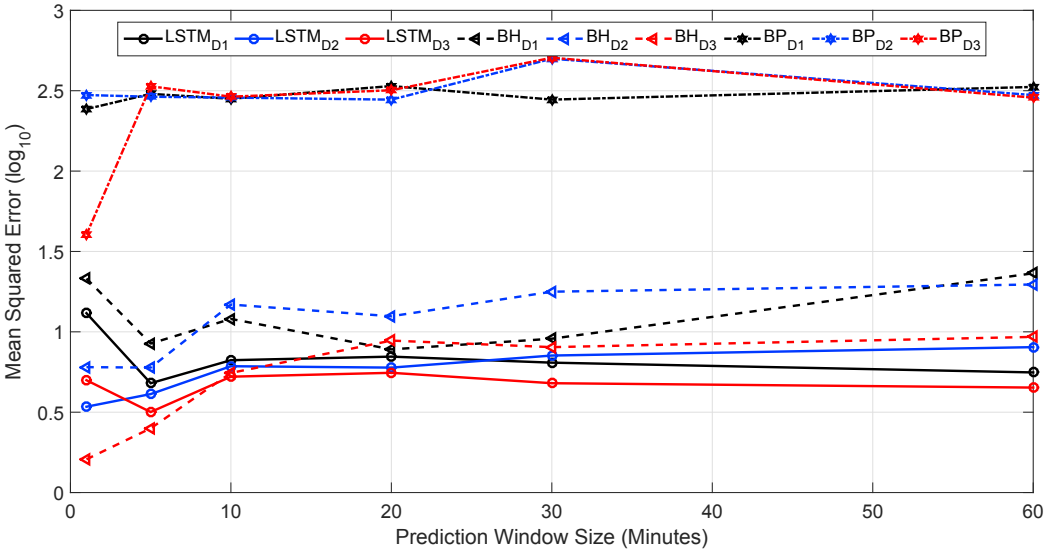


Figure 5: Mean squared errors

Table 1 shows the obtained mean squared error (in the order of 10^{-3}) over test data for all set of experiments. The minimum error achieved for D_1, D_2, and D_3 are 4.79×10^{-3}, 3.42×10^{-3}, and 3.17×10^{-3} respectively. Further, we compared the accuracy of proposed model with prediction methods based on black hole and back propagation learning algorithms [16, 20]. The graphical representation of the results and comparison is depicted in Fig. 5. The outcomes of experiments clearly convey that the LSTM-RNN based forecasting model outperforms both approaches.

5. Conclusions & Future Work

Efficient use of cloud resources is essential to gain most from cloud technology and it can be achieved by keeping the resources busy all the time. For a cloud server, it can not be possible to always have enough amount of work to keep all resources busy. In such scenarios workload prediction becomes helpful in resource scaling decisions as and

when required. The results of proposed workload forecasting method are prompting and encourage us to explore the domain further. Based on the empirical results, we conclude that an accurate workload prediction model does not only help in smart resource scaling decisions but also helps in promoting green computing by reducing the number of active machines. In future, we would like to develop a model based on different attributes of workloads.

Acknowledgment

The authors would like to thank Ministry of Electronics & Information Technology (MeitY), Govt. of India for providing financial support to carry out this work under Visvesvaraya PhD Scheme for Electronics & IT. Authors are also thankful to the anonymous reviewers for their constructive comments and suggestions to improve the manuscript.

References

[1] Columbus L. Roundup Of Cloud Computing Forecasts And Market Estimates; 2017. Available from: https://www.forbes.com/sites/louiscolumbus/2017/04/29/roundup-of-cloud-computing-forecasts-2017/#4b9c719431e8.
[2] Elsaid ME, Meinel C. Friendship based storage allocation for online social networks cloud computing. In: 2015 International Conference on Cloud Technologies and Applications (CloudTech); 2015. p. 1–6.
[3] Sharma MK, Vaisla KS. Towards Cloud Supported E-Governance Services Delivery Model. In: 2014 Fourth International Conference on Communication Systems and Network Technologies; 2014. p. 537–539.
[4] Mathew S. Implementation of Cloud Computing in Education - A Revolution. International Journal of Computer Theory and Engineering. 2012 June;4(3).
[5] Bednarzewska K, Pastuszak Z. CLOUD COMPUTING IN BUSINESS PROCESSES. A CAR PARTS SUPPLYING COMPANY CASE. In: Human Capital without Borders: Knowledge and Learning for Quality of Life; Proceedings of the Management, Knowledge and Learning International Conference; 2014. p. 1335–1343.
[6] Taneja H, Kapil, Singh AK. Preserving Privacy of Patients Based on Re-identification Risk. Procedia Computer Science. 2015;70(Supplement C):448 – 454. Proc. of the 4th Int. Conference on Eco-friendly Computing and Communication Systems.
[7] Kalekar PS, Rekhi K. Time series Forecasting using Holt-Winters Exponential Smoothing; 2004. .
[8] Li Q, Hao Qf, Xiao Lm, Li Zj. An Integrated Approach to Automatic Management of Virtualized Resources in Cloud Environments. Comput J. 2011 Jun;54(6):905–919.
[9] Sun YS, Chen YF, Chen MC. A Workload Analysis of Live Event Broadcast Service in Cloud. Procedia Computer Science. 2013;19:1028 – 1033. The 4th International Conference on Ambient Systems, Networks and Technologies (ANT 2013), the 3rd International Conference on Sustainable Energy Information Technology (SEIT-2013).
[10] Vercauteren T, Aggarwal P, Wang X, h Li T. Hierarchical Forecasting of Web Server Workload Using Sequential Monte Carlo Training. In: 2006 40th Annual Conference on Information Sciences and Systems; 2006. p. 899–904.
[11] Ardagna D, Casolari S, Colajanni M, Panicucci B. Dual time-scale distributed capacity allocation and load redirect algorithms for cloud systems. Journal of Parallel and Distributed Computing. 2012;72(6):796 – 808.
[12] Roy N, Dubey A, Gokhale A. Efficient Autoscaling in the Cloud Using Predictive Models for Workload Forecasting. In: Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing. CLOUD '11. Washington, DC, USA: IEEE Computer Society; 2011. p. 500–507.
[13] Cao L. Support vector machines experts for time series forecasting. Neurocomputing. 2003;51:321 – 339.
[14] Ban T, Zhang R, Pang S, Sarrafzadeh A, Inoue D. In: Lee M, Hirose A, Hou ZG, Kil RM, editors. Referential kNN Regression for Financial Time Series Forecasting. Berlin, Heidelberg: Springer Berlin Heidelberg; 2013. p. 601–608.
[15] Eddahech A, Chtourou S, Chtourou M. Hierarchical neural networks based prediction and control of dynamic reconfiguration for multilevel embedded systems. Journal of Systems Architecture. 2013;59(1):48 – 59.
[16] Kumar J, Singh AK. 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); 2016. p. 63–67.
[17] Kumar J, Singh AK. Workload Prediction in Cloud using Artificial Neural Network and Adaptive Differential Evolution. Future Generation Computer Systems. 2017;doi: https://doi.org/10.1016/j.future.2017.10.047 (In Press).
[18] Hochreiter S, Schmidhuber J. Long Short-Term Memory. Neural Comput. 1997 Nov;9(8):1735–1780. Available from: http://dx.doi.org/10.1162/neco.1997.9.8.1735.
[19] Arlitt MF, Williamson CL. Web Server Workload Characterization: The Search for Invariants. SIGMETRICS Perform Eval Rev. 1996 May;24(1):126–137. Available from: http://doi.acm.org/10.1145/233008.233034.
[20] Prevost JJ, 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; 2011. p. 276–281.