INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT (IJSREM)
VOLUME: 07 ISSUE: 08 | AUGUST - 2023 SJIF RATING: 8.176 ISSN: 2582-3930

Predicting Cloud Workload using Deep Neural Networks

Sonalee Sunhare[1] , Prof. Pankaj Raghuwanshi[2]
Department of CSE, AIT, Ujjain[1,2]

Abstract*—Cloud Computing has revolutionized computing off late with several domains and applications resorting to the cloud architecture. However effective task scheduling and load balancing is critical for cloud based servers. This is typically a very challenging task keeping in mind the fact that cloud workload is a parameter that depends on several other parameters. . Forecasting future workloads with high accuracy is especially challenging due to the randomness of the cloud workloads and also the non-deterministic nature of the governing or affecting parameters. Hence, due to the size and complexity of the data involved, finding regular patterns is a challenging task at hand. The present work proposes a back propagation based deep neural network architecture for cloud workload forecasting. The experiment uses the NASA cloud data set. The performance evaluation parameters have been chosen as mean absolute percentage error (MAPE) and regression. It has been found that the proposed system attains lesser mean square percentage error compared to previously existing technique [1].

Keywords*—Cloud Workload Estimation, Deep Neural Network (DNN), Principal Component Analysis (PCA), Steepest Descent Approach, Mean Absolute Percentage Error (MAPE).

I. INTRODUCTION

Cloud Computing has become one of the most sought after technologies which plays a pivotal role in several domains resorting to the high levels of data complexity, complex computation or applications needing hybrid platforms [1]-[2]. One of the most important aspects of cloud systems management is the fact that cloud servers sporadically face sudden surges in the number of requests often termed as cloud workload. This workload, if unforeseen can result in crash of the cloud server if alternate provisions are not made to handle the cloud workload [3]-[5]. This in term needs the estimate of cloud workloads in advance considering several governing factors. This is majorly critical especially for applications such as e-commerce and finance which may see sudden surges in requests. Thus there is a clear necessity of cloud workload prediction using models which can estimate cloud workloads with high accuracy. Statistical techniques are not found to be as accurate as the contemporary artificial intelligence and machine learning based approaches [6]. In this paper, a back propagation based approach for estimating cloud workload is proposed using deep learning architecture [7].

II. NEURAL NETWORKS

Deep learning has evolved as one of the most effective machine learning techniques which has the capability to handle extremely large and complex datasets [8]. It is training neural networks which have multiple hidden layers as compared to the single hidden layer neural network architectures [9]-[10].

The architectural view of a deep neural network is shown in figure 1. In this case, the outputs of each individual hidden layer is fed as the input to the subsequent hidden layer. The weight adaptation however can follow the training rule decided for the neural architecture. There are various configurations of hidden layers which can be the feed forward, recurrent or back propagation etc.

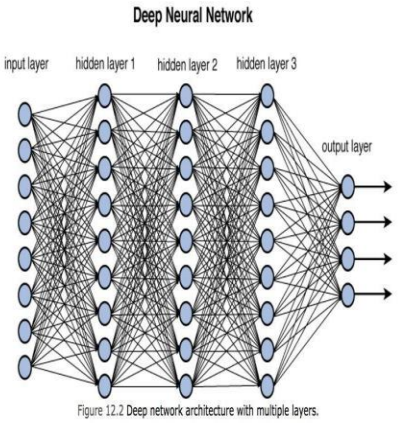

Fig.1 The Deep Neural Network Architecture

The figure above depicts the deep neural network architecture with multiple hidden layers. The output of the neural network however follows the following ANN rule:

$$Y = \sum_{i=1}^{n} X_i . W_i + \theta_i \quad (1)$$

Where,
X are the inputs

Y is the output
W are the weights
Θ is the bias.
Training of ANN is of major importance before it can be used to predict the outcome of the data inputs.

III. BACK PROP

Back propagation is one of the most effective ways to implement the deep neural networks with the following conditions:

1) Time series behavior of the data
2) Multi-variate data sets
3) Highly uncorrelated nature of input vectors

The essence of the back propagation based approach is the fact that the errors of each iteration is fed as the input to the next iteration. [11] –[13]. The error feedback mechanism generally is well suited to time series problems in which the dependent variable is primarily a function of time along with associated variables. Mathematically,

$$\boldsymbol{Y = f(t, V_1 \ldots . V_n)} \quad \textbf{(2)}$$

Here,
Y is the dependent variable
f stands for a function of
t is the time metric
V are the associated variables
n is the number of variables

The back propagation based approach can be illustrated graphically in figure 2.

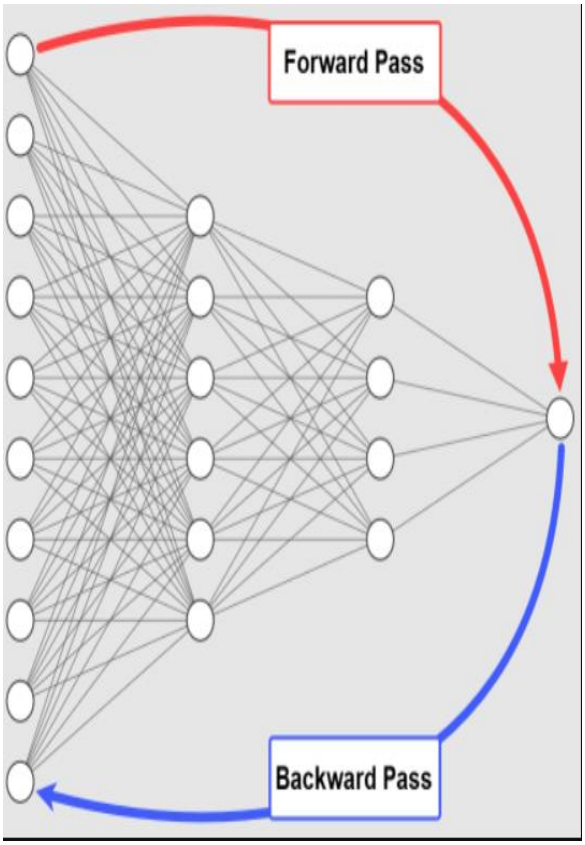

Fig.2 Error Feedback Mechanism

In case of back propagation, the weights of a subsequent iteration doesn't only depend on the conditions of that iteration but also on the weights and errors of the previous iteration mathematically given by:

$$\boldsymbol{W_{k+1} = f(W_k, e_k, V)} \quad \textbf{(2)}$$

Here,
W_{k+1} are the weights of a subsequent iteration
W_k are the weights of the present iteration
e_k is the present iteration error
V is the set of associated variables
In general, back propagation is able to minimize errors faster than feed forward networks, however at the cost of computational complexity at times. However, the trade off between the computational complexity and the performance can be clearly justified for large, complex and uncorrelated datasets for cloud data sets [14]-[15].

IV. GRADIENT DESCENT BASED TRAINING

The gradient descent algorithms (GDAs) generally exhibit:

1) Relatively lesser memory requirement
2) Relatively faster convergence rate

The essence of this approach is the updating of the gradient vector g, in such as way that it reduces the errors with respect to weights in the fastest manner. Mathematically, let the gradient be represented by g and the descent search vector by p, then

$$\boldsymbol{p_0 = -g_0} \quad \textbf{(3)}$$

Where,

g_0 denotes the gradient given by $\frac{\partial e}{\partial w}$

The sub-script 0 represents the starting iteration

The negative sign indicates a reduction in the errors w.r.t. weights

The tradeoff between the speed and accuracy is clearly given by the following relations:

$$\boldsymbol{W_{k+1} = W_k - \alpha g_x}, \ \boldsymbol{\alpha = \frac{1}{\mu}} \quad \textbf{(4)}$$

Here,

w_{k+1} is the weight of the next iteration

w_k is the weight of the present iteration

g_x is the gradient vector

μ is the step size for weight adjustment in each iteration.

The above equation shows stability in errors with monotonic decrease but needs higher number of iterations, specifically more in deep learning architectures due to direct computation of the Hessian Matrix of gradients. A faster approach is given by:

$$\boldsymbol{W_{k+1} = W_k - [J_K{}^T J_k]^{-1} J_K{}^T e_k} \quad \textbf{(5)}$$

In this case, the number of iterations reduce at the cost of the stable monotonic reduction of the errors with respect to weights.

Here,

J_k represents the Jacobian Matrix given by $\frac{\partial^2 e}{\partial_w{}^2}$

J_k^T represents the transpose of the Jacobian Matrix.

The speed of convergence is due to the indirect computation of the Hessian Matrix by using the Jacobian computation given by:

$$\mathbf{H = J_k{}^T J_k} \quad \textbf{(6)}$$

And

$$\mathbf{g = J_k{}^T e} \quad \textbf{(7)}$$

Here,

H is the Hessian Matrix

Finally, the GDA with both speed and stability optimized is given by:

$$\mathbf{W_{k+1} = W_k - \left[J_K{}^T J_k + \mu I\right]^{-1} J_K{}^T e_k} \quad \textbf{(8)}$$

Here,

The differentiating factor is the combination co-efficient μ which optimizes the GDA by adjusting the weights and thus the gradient.

K is the sub-script representing the iteration number

The activation function used for the algorithm is the tan-sig function mathematically defined as:

$$\boldsymbol{tansig(x) = \frac{2}{1+e^{-2x}} - 1} \quad \textbf{(9)}$$

V. PRINCIPAL COMPONENT ANALYSIS

The Principal Component Analysis is an optimization tool for the purpose of dimensional reduction of the data set. Consider a data set X having N samples. Out of the N sample, M samples may be highly correlated and hence may render low or little additional information to the training data.

$$\boldsymbol{M \underset{\in}{\rightarrow} N(X)} \quad \textbf{(10)}$$

Here,

M are the correlated samples

N are the total samples

X is the data set

If M samples are removed form the original data set, then there will be dimensional reduction in the data given by:

$$\boldsymbol{Y = X - M} \quad \textbf{(11)}$$

Here,

Y is the dimensionally reduced data set for more effective training.

VI. SYSTEM DESIGN

This input parameters used are:[1]
1) No. of servers
2) No. of users
3) Response time
4) Deviation delay value
5) Cloud Storage value
6) Mean Deviation value
7) Job Queueing value
8) Number of Operational Nodes
9) No. of Requests

The flowchart illustrates the summary of the system design.

The data is divided in the ration of 70:30 for training and testing data set bifurcation.

The final performance metrics computed for system evaluation are:

1) Mean Absolute Percentage Error (MAPE)

$$MAPE = \frac{100}{M}\sum_{t=1}^{N}\frac{E-E_t|}{E_t} \qquad (12)$$

Here E_t and E_t˜ stand for the predicted and actual values respectively.

The number of predicted samples is indicated by M.

2) Regression

The extent of similarity between two variables is given by the regression where the maximum value is 1 and the minimum is 0.

VII. RESULTS

The results have been evaluated based on the following parameters:

1. (MAPE)
2. Regression
3. MSE w.r.t. the number of epochs

Fig.4 Designed Neural Network

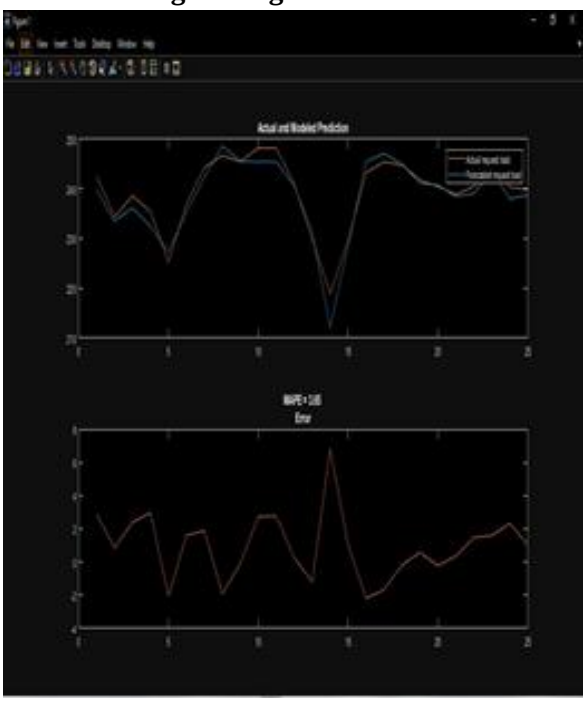

Fig.5 Predicted and Actual Cloud Workload

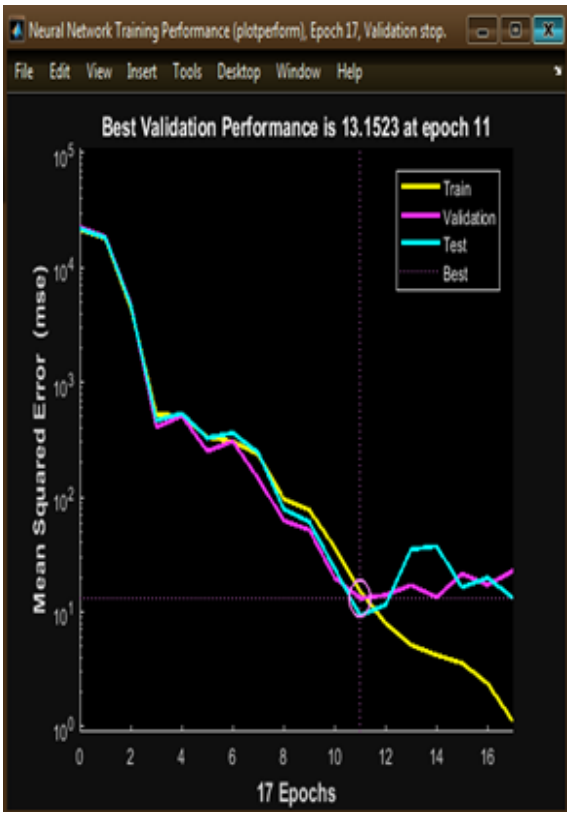

Fig.6 Variation of MSE with respect to epochs

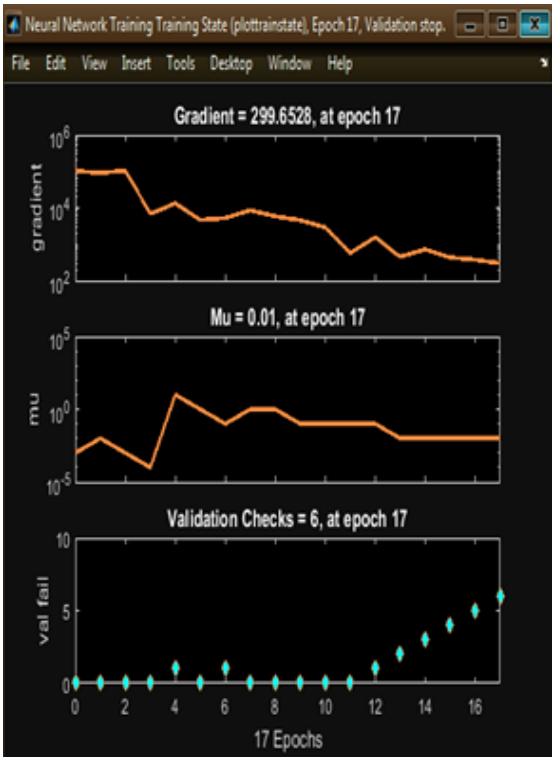

Fig.7 Training Parameters

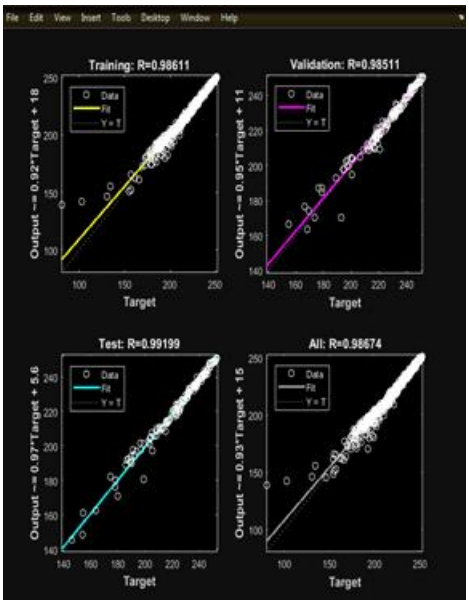

Fig.8 Regression Analysis

From the above figures, it can be concluded that the proposed system attains the following results:

1) MAPE of 3.65%
2) Regression of 0.98 (overall)
3) Number of iterations is 17

A comparative accuracy analysis w.r.t. previous work is given by:

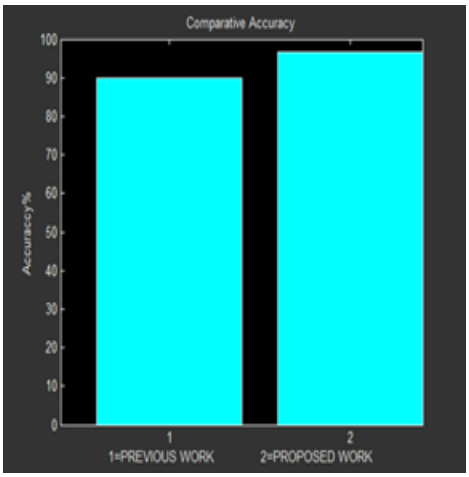

Fig.9 Comparative Accuracy Analysis w.r.t. Previous Work [1]

CONCLUSION

It can be concluded from the previous discussions that cloud workload estimation is critical for real time applications which use the cloud architecture. However, the cloud workload is sporadic and random in nature due to the large number of governing uncorrelated variables. Hence estimating cloud workloads with high accuracy is challenging. In the proposed approach a back propagation based deep learning model is proposed with a 1-10-1 configuration. The adaptive gradient descent algorithm (GDA) is used to train the neural network. It has been shown that the proposed work attains a mean absolute percentage error of 3.65% compared to a mean absolute percentage error of 10.26% of previous work [1]. Moreover, the regression is 0.98 at the number of epochs being 17. Thus the proposed system is able to achieve low errors, higher accuracy and relatively low number of iterations.

REFERENCES

[1] P Yazdanian, S Sharifian, E2LG: a multiscale ensemble of LSTM/GAN deep learning architecture for multistep-ahead cloud workload prediction", Journal of Supercomputing, Springer 2022, vol. 77, pp.11052–11082.

[2] J. Gao, H. Wang and H. Shen, "Machine Learning Based Workload Prediction in Cloud Computing," 2020 29th International Conference on Computer Communications and Networks (ICCCN), 2021, pp. 1-9

[3] Z. Chen, J. Hu, G. Min, A. Y. Zomaya and T. El-Ghazawi, "Towards Accurate Prediction for High-Dimensional and Highly-Variable Cloud Workloads with Deep Learning," in IEEE Transactions on Parallel and Distributed Systems, 2020, vol. 31, no. 4, pp. 923-934.

[4] L. Wang and E. Gelenbe, "Adaptive Dispatching of Tasks in the Cloud," in IEEE Transactions on Cloud Computing, vol. 6, no. 1, pp. 33-45, 1 Jan.-March 2018

[5] Martin Duggan, Karl Mason, Jim Duggan, Enda Howley, Enda Barrett, "Predicting Host CPU Utilization in Cloud Computing using Recurrent Neural Networks", 2017 IEEE.

[6] Ning Liu, Zhe Li, Jielong Xu, Zhiyuan Xu, Sheng Lin, Qinru Qiu, Jian Tang, Yanzhi Wang, "A Hierarchical Framework of Cloud Resource Allocation and Power Management Using Deep Reinforcement Learning", 2017 IEEE.

[7] Liyun Zuo, Shoubin Dong, Lei Shu, Senior Member, IEEE, Chunsheng Zhu, Student Member, IEEE, and Guangjie Han, Member, IEEE, "A Multiqueue Interlacing Peak Scheduling Method Based on Tasks' Classification in Cloud Computing", 2016 IEEE.

[8] Yazhou Hu, Bo Deng, Fuyang Peng and Dongxia Wang, "Workload Prediction for Cloud Computing Elasticity Mechanism", 2016 IEEE.

[9] Ji Xue, Feng Yan, Robert Birke, Lydia Y. Chen, Thomas Scherer, and Evgenia Smirni, "PRACTISE: Robust Prediction of Data Center Time Series", 2015 IEEE.

[10] Mehmet Demirci, "A Survey of Machine Learning Applications for Energy-Efficient Resource Management in Cloud Computing Environments", 2015 IEEE.

[11] Sherif Abdelwahab, Member, IEEE, Bechir Hamdaoui, Senior Member, IEEE, Mohsen Guizani, Fellow, IEEE, and Ammar Rayes, "Enabling Smart Cloud Services Through Remote Sensing: An Internet of Everything Enabler", 2014 IEEE.

[12] Chin-Feng Lai, Member, IEEE, Min Chen, Senior Member, IEEE, Jeng-Shyang Pan, Chan-Hyun Youn, Member, IEEE, and Han-Chieh Chao, Senior Member, IEEE, "A Collaborative Computing Framework of Cloud Network and WBSN Applied to Fall Detection and 3-D Motion Reconstruction", 2014 IEEE.

[13] Ian Davis, Hadi Hemmati, Ric Holt, Mike Godfrey, Douglas Neuse, Serge Mankovskii, "Storm Prediction in a Cloud", 2013 IEEE.

[14] Abul Bashar, "Autonomic Scaling of Cloud Computing Resources using BN-based Prediction Models", 2013 IEEE.

[15] Sadeka Islam , Jacky Keunga, Kevin Lee, Anna Liu, "Autonomic Scaling of Cloud Computing Resources using BN-based Prediction Models", 2012 ELSEVIER.

[16] Erol Gelenbe, Ricardo Lent and Markos Douratsos, "Choosing a Local or Remote Cloud", 2012 IEEE.

[17] Mohammad Moein Taheri and Kamran Zamanifar, "2-Phase Optimization Method for Energy

Aware Scheduling of Virtual Machines in Cloud Data Centers", 2011 IEEE.

[18] Saurabh Kumar Garg, Srinivasa K. Gopalaiyengar, and Rajkumar Buyya, "SLA-Based Resource Provisioning for Heterogeneous Workloads in a Virtualized Cloud Datacenter", 2011 IEEE.

[19] Md. Toukir Imamt, Sheikh Faisal Miskhatt, Rashedur M Rahmant, M. Ashraful Amin, "Neural Network and Regression Based Processor Load Prediction for Efficient Scaling of Grid and Cloud Resources", 2011 IEEE.