A Deep Learning Approach for VM Workload Prediction in the Cloud

Feng Qiu[1]
[1]Software College
Northeastern University
Shenyang, China
huatang@yeah.net

Bin Zhang[2], Jun Guo[2]
[2]School of Computer Science and Engineering
Northeastern University
Shenyang, China
zhangbin@mail.neu.edu.cn
boatheader@163.com

***Abstract*—In order to manage the resources in cloud efficiently, ensure the performance of cloud services and reduce the power consumption, it is critical to predict the workload of virtual machines (VM) accurately. In this paper, a new approach for VM workload prediction based on deep learning was proposed. A deep learning prediction model was designed with a deep belief network (DBN) composed of multiple-layered restricted Boltzmann machines (RBMs) and a regression layer. The DBN is used to extract the high level features from all VMs workload data and the regression layer is used to predict the workload of the VMs in the future. With little prior knowledge, DBN could learn the features efficiently for the VM workload prediction in an unsupervised fashion. Experimental results show that the proposed approach improves the workload prediction performance compared with other widely used workload prediction approaches.**

Keywords—deep learning; deep belief networks; restricted Boltzmann machine; workload prediction

I. INTRODUCTION

Cloud computing is the next major paradigm shift in information and communication technology (ICT) [1]. Cloud computing leverages the virtualization technology to host a variety of applications or services instances on virtual machines (VMs). Moreover, multiple heterogeneous VMs can be deployed in the same physical machine (PM). A huge shared resource pool constructed with software and hardware is provided in a pay-as-you-go model to the cloud customers. In a cloud, the VM workload influences resource provisioning, quality of service (QoS) guarantee and energy consumption. If the VM were overload, cloud provider would reallocate more resources and avoid the service level agreement (SLA) violation. On the contrary, the VM would scale down when the allocated resources are under-utilized. Therefore, forecasting the VM workload accurately is essential to enhance the utilization of cloud computing resources, ensure the availability of cloud services and reduce the consumption of energy.

Cloud computing is a cluster of numerous PMs that are used to deploy multiple VMs. Nowadays, most cloud applications are composed of a set of cooperating service components with different functionality. These service components are running across multiple heterogeneous VMs that are not independent. For example, a typical web application includes three tiers: presentation tier, application tier and data tier, implemented as web server, application server and database server, respectively [2]. If the web application is deployed in the cloud, each tier of the application would be hosted in one or more VMs and interact with other tiers that are hosted in different VMs. Hence, the cloud system is a complex network of the mass correlative VMs. The workload on these VMs are often driven by repeatable behavior of users' requests and hence, exhibit temporal and spatial correlations [3]. From the temporal aspect, the workload on VM caused by users' requests may fluctuate with periodic pattern. From the spatial aspect, there is correlation between the VMs that run the one of tiers of one web application. If the workload on VM hosted with web server increases caused by the incoming users' requests, it will lead to the increase of workload on VMs hosted with application server and database server. Therefore, we propose a workload prediction approach based on the workload datasets of all VMs instead of one individual VM in the cloud.

There have been many proposals for VM workload prediction approaches. Zhen *et al.* [4] propose a shot-term prediction method to predict the CPU load based on exponentially weighted moving average (EWMA). Kousiouris *et al.* [5] propose an application workload prediction method based on the artificial neural networks (ANNs) optimized by genetic algorithm. Islam *et al.* [6] apply the ANNs and linear regression for prediction of resources required for applications. Calheiros *et al.* [7] propose a workload prediction module to estimate the demanded resources and allocate or release them in advance. However, there are three defects of the above approaches. First, most of them employ shallow structure. Some prediction approaches, such as the Autoregressive Integrated Moving Average (ARIMA) model, employ the linear prediction structure. The approach based on Neural Network has only one hidden layer in the structure. Because of the stochastic and nonlinear nature of the workload data, the prediction approaches based on shallow structure have poor ability to discover the inherent features in the workload data. Second, almost all of the existing approaches predict the VM workload in the cloud respectively. The VM workload prediction is based on its own workload historical trace, regardless of the correlation among VMs. It neglects that the cloud computing system is actually a complex network consists of numerous VMs, which possess the temporal and spatial

978-1-5090-2239-7/16/$31.00 copyright 2016 IEEE
SNPD 2016, May 30-June 1, 2016, Shanghai, China

correlation. Finally, the monitored workload data sets consist of high veracity data. That means, there are a large number of inaccurate and incomplete data in the workload data sets. It is likely to result in the inaccurate prediction based on only one VM workload trace.

To address these problems, we propose a VM workload prediction approach based on deep learning. The proposed approach presents a deep learning model that can learn the inherent features from the workload data of all VMs in the cloud. The high level features that are learned with deep learning model are more effective for the workload prediction than the raw workload data with noise [8]. Then the learned features can be used to predict the VM workload in the future. The ensemble architecture is designed for the workload prediction with two parts. First, we use DBN as a deep learning model to learn the high-level features of the historical workload data in an unsupervised manner. DBN, the most common and effective deep learning model, has remarkable power to deal with the prediction problem with the complex correlations among the numerous VMs and a large amount of workload trace data. In addition, our approach only needs little prior knowledge to extract the representative features. Based on this deep learning architecture, a regression layer will be added that performs the fine-tuning of the whole model and the workload prediction in supervised learning. In general, several metrics can be regarded as the measurements of VM workload, such as CPU utilization, memory utilization, etc. In this paper, we only discuss CPU utilization. But our approach can be applied to other resource metrics respectively or the set of the metrics. Although deep learning need more computation and storage costs, we use the graphic processing unit (GPU) in computer to boost the processing speed. The experiment results show that our approach can improve the workload prediction performance compared with other prediction approaches.

The remainder of this paper is structured as follows. In section II, we discuss the VM workload prediction problem in cloud computing and research background knowledge of DBN. In section III, we present our workload prediction approach based on the deep learning model. In section IV, we present the results of experiments and the comparison between our approach and other widely used approaches. Finally, the conclusion and the future work are given in section V.

II. Poblem Statement and Research Background

A. VM Workload Prediction

The workload prediction has important practical significance for cloud providers. When the enterprise users deploy their applications in the cloud, the service components of the application will be hosted in one or more VMs. The enterprise users and the cloud providers usually sign a Service Level agreement (SLA). Therefore, the latter must supply adequate resources to the applications to ensure that the SLA violation would be avoided. In general, the cloud providers allocate the initial resources to every VM hosted by the service. But the VM workload will drastic fluctuate with the variation of the service access requirements of the cloud end-users. On the one hand, the quality of services may experience degradation when the workload increases suddenly or persistently over time; on the other hand, the reduction of the workload is likely to cause the low utilization of the allocated resources. The cloud system should have an ability to manage the resources to enhance the resource utilization and reduce the operation cost. There are two ways to manage the resources: reactive resource management and proactive resource management [9]. The reactive resource management is only based on the monitored workload data and would lead to the time-lag effect for the resources adjustment. Because the cloud system need enough time to manage the resources, it is better to estimate the future workload beforehand. Therefore, the proactive resource management is more suitable for cloud computing system. According to the accurate prediction of the workload, the resources of VM would be auto-scaling in advance on-demand to guarantee the quality of service and reduce the consumption of energy.

The state-of-the-art workload prediction approaches, such as ARIMA, are all most based on the statistical analysis of the workload trace of one single VM. The prediction model would be designed to capture the workload variation patterns and inherent features according to its own workload trace of one VM to predict its workload in the future. Workload can be classified according to the fundamental resources consumed in terms of CPU, memory and so on, or service requests rate, latency sensitive and batch workloads depending on the interaction with the end-users [7]. We assume that the workload trace is a workload time series, that can be defined as $X = (x_{i,1}, x_{i,2}, x_{i,3}, \dots, x_{i,j})$, where $i = 1,2,3, \dots, S$ and $j = 1,2,3, \dots, T$. S is the number of all VMs in the cloud and T is the number of the time intervals that have been observed. The $x_{i,j}$ is the workload of the *i*th VM at the *j*th time interval. The prediction problem is that according to the previous *m* selected workload time series $X_{select} = (x_{i,T-m+1}, \dots, x_{i,T-1}, x_{i,T})$, a prediction model is designed to predict the workload $X_{i,T+n}$ of the *i*th VM at time interval $(T + n)$ for *n* prediction intervals. The prediction model should have good performance to achieve that the predicted workload be close to the actual workload as far as possible. The evaluation of the prediction model is based on the mean absolute percentage error (MAPE) defined as:

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|x_i^a - x_i^p|}{x_i^a} \quad (1)$$

where x_i^a is the actual workload value and x_i^p is the estimated workload.

B. Deep Belief Network

Deep Belief Network (DBN) is one of the most common and effective deep learning models proposed by G .Hinton in 2006 [10]. It has been widely adopted for many studies and practical applications, such as image classification, speech recognition and audio classification. DBN is also the novel efficient machine learning approach for many prediction tasks [11], [12]. DBN is a deep neural network that is composed of the stacked Restricted Boltzmann Machine (RBM). In DBN, the hidden layer of one RBM is used as the visible layer of next RBM in the stack. Hinton *et al.* proposed a fast unsupervised greedy layerwise training method to train DBN [10].

RBM (See Fig.1) is derived from Boltzmann Machine (BM) which is composed of only one visible layer (v) and one hidden layer (h). But instead of the full connections of the neuron units in BM, the neuron units within the visible layer or the hidden layer have no connections in RBM. In RBM, the visible layer is used as input layer of data and the hidden layer is used as output layer for features. w is the weight matrix between the units of visible layer and the hidden layer. In order to obtain the connection weight matrix rapidly, Hinton has proposed the Contrastive Divergence (CD) method in 2002 [13].

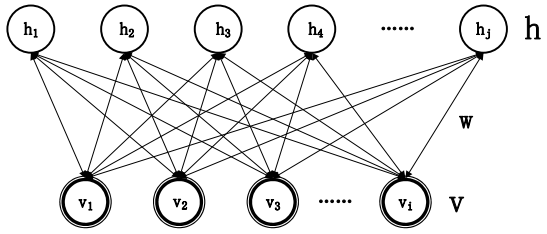

Fig.1. Restricted Boltzmann Machine (RBM)

The typical RBM is composed of the stochastic binary units in the visible and hidden layer. But the unit can be any exponential family unit, such as Softmax unit, Gaussian unit and Poisson unit, etc. [14]. Here we assume all visible and hidden units are binary value. RBM is an energy-based model. The energy function can be written as

$$E(v,h|\theta) = -\sum_{i=1}^{V} a_i v_i - \sum_{j=1}^{H} b_j h_j - \sum_{i=1}^{V}\sum_{j=1}^{H} v_i w_{ij} h_j \quad (3)$$

with parameters $\theta = (w, a, b)$, w_{ij} is the connection weight between the visible unit v_i and hidden unit h_j, and a_i and b_j is the bias of v_i and h_j, respectively. The number of visible units and hidden units is V and H respectively. The joint probability distributions of the visible-hidden vectors pair can be written as

$$p_\theta(v,h) = \frac{1}{Z} e^{-E(v,h)} \quad (4)$$

where Z is the partition function. When the visible vector $\boldsymbol{V}$ is fixed, then the probability that hidden unit h_j is activated is

$$p_\theta\left(h_j = 1 \middle| v\right) = \delta\left(\sum_{i=1}^{V} w_{ij} v_i + b_j\right) \quad (5)$$

Similarly, when the hidden vector H is fixed, the probability that visible unit $\mathrm{v_i}$ is activated is

$$p_\theta(v_i = 1|h) = \delta\left(\sum_{j=1}^{H} w_{ij} h_j + a_i\right) \quad (6)$$

where $\delta(x)$ is a activation function. $\delta(x)$ can be a logistic function, e.g. $\delta(x) = \frac{1}{1+exp(-x)}$. Due to the exponentially expensive computation during the training, the parameters $\theta = (w, a, b)$ are trained to minimize the reconstruction error using contrastive divergence [13]. The parameters are updated as follow:

$$\Delta w_{ij} = \varepsilon\left(\langle v_i h_j\rangle_{data} - \langle v_i h_j\rangle_{recon}\right) \quad (7)$$

$$\Delta a_j = \varepsilon(p(v_i)_{data} - p(v_i)_{recon}) \quad (8)$$

$$\Delta b_j = \varepsilon\left(p(h_j)_{data} - p(h_j)_{recon}\right) \quad (9)$$

where the ε is the learning rate, $\langle v_i h_j\rangle$ means the expectation of units on two layers. $\langle\cdot\rangle_{data}$ means the empirical expectation and $\langle\cdot\rangle_{recon}$ means the expectation that the model reconstruct the original visible vector [15].

As DBN is formed by the stacked RBMs, the training of DBN is repeated greedily with each RBM as the building block from the bottom layer to the highest layer [16]. The training procedure is as follow: the visible layer and the first hidden layer form a RBM that learns the high level features of the raw input data. At the same time, the parameters θ would been learned. The learned feature will be used as input data of the second RBM formed by the first hidden layer and the second hidden layer. Then we train the second RBM and obtain the high level features and the parameters θ of the second RBM again. We can continue training the hidden layers as many as we need. Unlike other models, DBN can learn the features in unsupervised fashion without the labeled data [10].

III. Deep Learning Model for Workload Prediction

As mentioned earlier, the cloud computing system is a parallel and distributed system and there are the complex correlations among the VMs in the cloud. Moreover, the workload data is stochastic and nonlinear. It is better to predict workload based on the high level features learned from the workload data of the correlative VMs than the original workload data of one single VM. The deep model can discover the implicit variation patterns and inherent features from the large number of workload data. Therefore, the deep learning model is more powerful in learning the features from the workload data with such a complex relations than the shallow model.

In this paper, the proposed deep learning model is composed of a DBN and a logistic regression layer. DBN can extract the high-level features from the row workload data of VMs in an unsupervised fashion. Essentially, DBN is a multilayer neural network. It becomes extremely difficult because of the error gradient would explode or vanish with the increase of number of hidden layers [16]. This problem has been solved by using the fast unsupervised greedy layerwise training method to train the deep neural networks [10]. In addition, DBN can perform the feature learning with little prior knowledge. The top layer above the DBN is a sigmoid regression layer to fine-tune the whole deep learning model in a supervised way. We use the CPU utilization data of all VMs of several previous time intervals as the input data of DBN and predict the CPU utilization of one single VM in the future interval.

The whole deep architecture of the VM workload prediction is showed in Fig.2. The input of the deep architecture is the all monitored VMs CPU utilization in several time intervals. The estimated CPU utilization of the VM is outputted in the top layer. The value of CPU utilization is normalized in the range [0, 1].The units' number in the first layer is the multiplication of the number of VMs (N_{vm}) in the

cloud and the number of the time intervals (N_{interv}), e.g. ($N_{vm} \times N_{interv}$). We can use the deep learning model to predict the workload of a single VM. It is also possible to predict the workload of multiple VMs at the same time. The procedures of feature learning for one VM and multiple VMs are same. The difference is that at the top layer: the logistic regression layer perform the fine-tuning of the whole structure using the workload data of all VMs to predict as the label data in supervised manner.

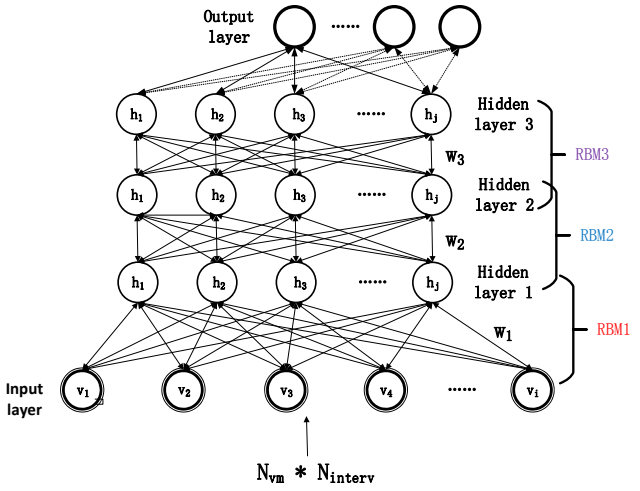

Fig.2. Deep architecture for the Workload Prediction of a single VM (connected with solid line in the top layer) and multiple VMs (connected with solid and dotted line in the top layer)

Because the input data of the deep architecture are real values, the typical binary RBM is replaced by Gaussian-Bernoulli RBM (CBRBM) [17], where the only change in the RBM energy function is to the visible units biases, by adding a bias term that is quadratic in the visible units [18]. The energy function and the conditional probability distribution can be expressed as:

$$E(v,h|\theta) = \sum_{i=1}^{V} \frac{(v_i - a_i)^2}{2\sigma_i^2} - \sum_{j=1}^{H} b_j h_j - \sum_{i=1}^{V}\sum_{j=1}^{H} \frac{v_i}{\sigma_i} h_j w_{ij} \quad (10)$$

$$p(h_i|v;\theta) = \delta\left(\sum_{i=1}^{V} w_{ij} v_i + b_j\right) \quad (11)$$

$$p(v_i|h;\theta) = N\left(\sigma_i \sum_{j=1}^{H} w_{ij} h_j + a_i, \sigma_i^2\right) \quad (12)$$

Here, σ is the standard deviation vector of the Gaussian visible units and $N(\mu, \sigma^2)$ is the probability distribution function of the Gaussian distribution with mean μ and variance σ^2.

According to the training method proposed by Hinton, the prediction process is described as follow:

Step 1: train the visible layer and first hidden layer as a RBM in an unsupervised fashion. The part of the workload dataset is the input of the RBM. Then the parameters θ of RBM are kept.

Step 2: the hidden layer of the first RBM becomes the visible layer of the next RBM. The extracted feature of the first RBM is the input of the next RBM. Then train the next RBM. The parameters 0 of the next RBM is kept as well as the first RBM.

Step 3: repeat the step 2 for the predefined number of the hidden layers.

Step 4: use the output of the last RBM as the input of the logistic regression layer. The train the regression layer with some labeled data in a supervised way.

Step 5: use the BP algorithm to fine-tune the whole parameters 0 that are kept before by supervised training.

IV. EXPERIMENT

A. Experiment dataset and Evaluation Methods

In this paper, we discuss only the prediction of the CPU utilization of the VMs. The experimental workload data sets are provided from the platform *PlanetLab* included in the simulation tool *Cloudsim* [19]. The *PlanetLab* is a open platform for developing , deploying and accessing planetary-scale services. Many academic institutes and industrial research labs have used *PlanetLab* to develop new technologies. In *Cloudsim*, there is a data set of *PlanetLab* that supplies the real VMs CPU utilization data. Over 1000 VMs have been monitored and the CPU utilization have been collected every 5 minutes.

In this paper, we use the CPU utilization of all VMs as the input data of the prediction model. We use the CPU utilization data of 8 days as the training data and the remaining two days data as testing data. Before the training and testing, all CPU utilization data were normalized into [0, 1]. We use the CPU utilization data of all VMs in a history window as the input vector of the deep learning model. We choose the 20 continuous running VMs in 10 days in *PlanetLab* as the VMs we need to predict. There are two prediction tasks: predicting the CPU utilizations of the VM with the high workload among these 20 VMs in one time interval and predicting the CPU utilization of all 20 VMs in several time intervals (5-minutes, 15-minutes and 30-minutes) in the future. As mentioned earlier, the performance of the prediction approaches is evaluated with MAPE. For the first workload prediction task, we use these two measurements directly. But for the second one, we will compute the average MAPE for all 20 VMs, which are presented as A-MAPE.

B. Deep Learning Structure

In order to build the deep learning model for workload prediction, it needs to determine some parameters of the deep learning structure at first, such as the number of the time intervals for training, the number of hidden layers, the number of units in each hidden layer and the number of training iterations. The number of the time intervals determines the window size of the historical CPU utilization data for the feature learning and the value from {6, 9, 12, 15, 18, 21, 24} (the length is from 0.5 hours to 2 hours, each time increased with 15 minutes). The number of hidden layers in the deep architecture is set from 1 to 7. For simplicity, we set all hidden layers with the same number of the neural units and the value from {10,20,30,40,50,60,70,80,90,100}. The other important

parameter is the iteration times of the model training. If the number is set too big, it is likely to result in the overfitting of the training data. Hence, the iteration times is set with the value from 10 to 70. We use the cross validation method to find out the best parameters sets. The deep learning structures for different prediction tasks are given in Table I.

TABLE I. DEEP LEARNING STRUCTURES FOR WORKLOAD PREDICTION

Task	Time intervals	Hidden layers	Hidden units	Iteration
5-min	12	3	50	50
15-min	18	4	60	60
30-min	21	4	80	50

C. *Results and Analysis*

In this section, we first present the prediction performance of our approach. For the first prediction task, the CPU utilization prediction of the VM with the high workload in one testing day is illustrated in Fig.3. For this prediction task, the CPU utilization is predicted in one time interval (5 minutes) in the future. In Fig.3, the proposed prediction model shows good performance for the workload prediction. It can be seen that the predicted CPU utilization data are close to the monitored CPU utilizations. And the predicted variation trend is also similar the real situation. Although there are some deviations between the predicted workload and actual workload, the proposed prediction model shows still the good generalization ability with the small training sets and many bursty workloads.

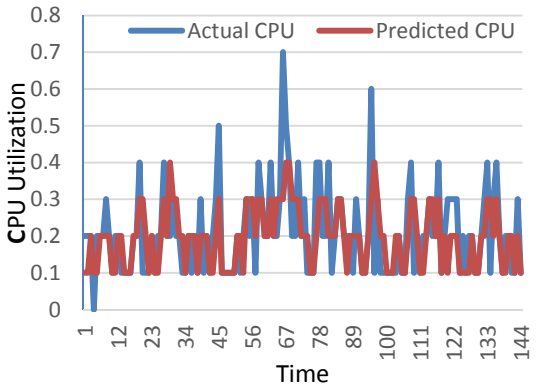

Fig. 3. Workload Prediction of one VM

Then we compare our prediction model with the EWMA model, the simple NN model with one hidden layer, the multilayer NN model that is same with the proposed DBN structure and the ARIMA model for single VM workload and multiple VMs workload prediction tasks. We use these prediction methods to predict the CPU utilization of the previous VM with the high workload and of the 20 VMs we choose. For the multiple VMs workload prediction task, we use the average MAPE (A-MAPE) as the performance index.

Fig.4 presents the performance comparison of these prediction approaches for the single VM workload prediction. Because the prediction time interval is short, EWMA model and ARIMA model display good prediction performance as well as the proposed deep prediction model. The proposed deep prediction model can only improve the performance with 1.3%. But we can also find that the proposed prediction model has more significant advantage for the multiple VMs workload prediction in several time intervals. Fig. 5 shows that, for the long-term workload prediction, the proposed deep prediction model can improve the accuracy with the representative feature learning from the workload data of all correlative VMs and the unsupervised learning manner. The prediction accuracy would be increased more than 2.5%. Therefore, the proposed deep prediction model can use the predicted workload to support the cloud system to manage the resources in advance.

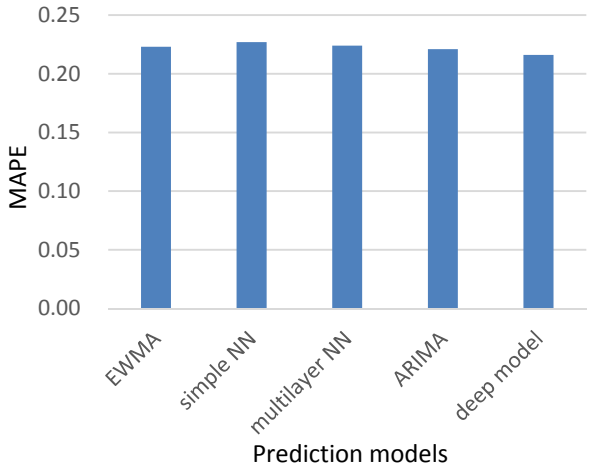

Fig. 4. Comparison of single VM workload prediction

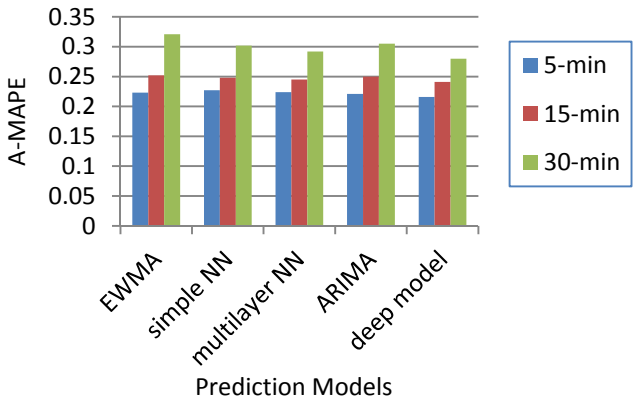

Fig. 5. Comparison of multiple VMs workload prediction in several time intervals

V. CONCLUSION

In this paper, we have presented a new deep learning approach for the VM workload prediction in the cloud system. A deep learning model is designed that consists of a DBN and a regression layer. With little prior knowledge, DBN learn a high level representation of features effectively in an

unsupervised fashion. The top regression layer is used to fine-tune the whole deep model in supervised manner and predict the CPU utilization of the VM in the future. Compared with other widely used prediction approaches, our approach uses the workload data of all VMs to perform the prediction. It is shown from the experimental results that the proposed prediction approach based on deep learning can improve the accuracy of the CPU utilization prediction than some of the existing prediction approaches. Our workload prediction approach has good prediction performance than other workload prediction methods. In future work, we will focus on the researches of the prediction of the multiple correlative VMs workloads considered the temporal and spatial correlations among the VMs at the same time. We also study on the efficient training of deep learning model and the rapid parameters determination of the deep architecture involved in the optimization of the deep learning method.

REFERENCES

[1] K. Bilal, S. U. R. Malik, S. U. Khan and A. Y. Zomaya. "Trends and challenges in cloud datacenters". IEEE Cloud Computing, 2014 (1): 10-20.

[2] D. Huang, B. He and C. Miao. "A survey of resource management in multi-tier web applications". Communications Surveys & Tutorials, IEEE, 2014, 16(3): 1574-1590.

[3] A. Khan, X. Yan, S. Tao and N. Anerousis. "Workload characterization and prediction in the cloud: A multiple time series approach". Network Operations and Management Symposium (NOMS), 2012 IEEE. IEEE, 2012: 1287-1294.

[4] Z. Xiao, W. Song and Q. Chen. "Dynamic resource allocation using virtual machines for cloud computing environment". Parallel and Distributed Systems, IEEE Transactions on, 2013, 24(6): 1107-1117.

[5] G. Kousiouris, A. Menychtas, D. Kyriazis, et al. "Parametric design and performance analysis of a decoupled service-oriented prediction framework based on embedded numerical software". Services Computing, IEEE Transactions on, 2013, 6(4): 511-524.

[6] S. Islam, J. Keung, K. Lee and A. Liu. "Empirical prediction models for adaptive resource provisioning in the cloud". Future Generation Computer Systems, 2012, 28(1): 155-162.

[7] R. 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, NO. 4, 449-458, OCTOBER-DECEMBER 2015

[8] Q. Yang, Y. Zhou, Y. Yu, J, Yuan, X. Xing and S. Du. "Multi-step-ahead host load prediction using autoencoder and echo state networks in cloud computing". The Journal of Supercomputing, 2015: 1-17.

[9] I. S. Moreno, P. Garraghan, P. Townend and J. Xu. "Analysis, modeling and simulation of workload patterns in a large-scale utility cloud". Cloud Computing, IEEE Transactions on, 2014, 2(2): 208-221.

[10] G. E. Hinton, S. Osindero and Y. W. Teh. "A fast learning algorithm for deep belief nets". Neural computation, 2006, 18(7): 1527-1554.

[11] T. Kuremoto, S. Kimura, K. Kobayashi and M. Obayasi. "Time series forecasting using a deep belief network with restricted Boltzmann machines". Neurocomputing, 2014, 137: 47-56.

[12] W. Huang, G. Song, H. Hong and K. Xie. "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning". Intelligent Transportation Systems, IEEE Transactions on, 2014, 15(5): 2191-2201.

[13] G. E. Hinton. "Training products of experts by minimizing contrastive divergence". Neural computation, 2002, 14(8): 1771-1800.

[14] M. Welling, M. Rosen-Zvi and G. E. Hinton. "Exponential family harmoniums with an application to information retrieval". Advances in neural information processing systems. 2004: 1481-1488.

[15] R. Sarikaya, G. E. Hinton and A. Deoras. "Application of deep belief networks for natural language understanding". Audio, Speech, and Language Processing, IEEE/ACM Transactions on, 2014, 22(4): 778-784.

[16] Y. Bengio. "Learning deep architectures for AI". Foundations and trends® in Machine Learning, 2009, 2(1): 1-127.

[17] G. E. Hinton and R. R. Salakhutdinov. "Reducing the dimensionality of data with neural networks". Science, 2006, 313(5786): 504-507.

[18] Y. Bengio, A. Courville and P. Vincent. "Representation learning: A review and new perspectives". Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2013, 35(8): 1798-1828.

[19] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. Rose and R. Buyya. "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms". Software: Practice and Experience, 2011, 41(1): 23-50.