Machine Status Prediction for Dynamic and Heterogenous Cloud Environment

Jinliang Xu, Shangguang Wang*, Ao Zhou, Fangchun Yang
State Key Laboratory of Networking and Switching Technology
Beijing University of Posts and Telecommunications, Beijing, China
{jlxu,sgwang,aozhou,fcyang}@bupt.edu.cn

***Abstract*—The widespread utilization of cloud computing services has brought in the emergence of cloud service reliability as an important issue for both cloud providers and users. To enhance cloud service reliability and reduce the subsequent losses, the future status of virtual machines should be monitored in real time and predicted before they crash. However, most existing methods ignore the following two characteristics of actual cloud environment, and will result in bad performance of status prediction: 1. cloud environment is dynamically changing; 2. cloud environment consists of many heterogeneous physical and virtual machines. In this paper, we investigate the predictive power of collected data from cloud environment, and propose a simple yet general machine learning model StaP to predict multiple machine status. We introduce the motivation, the model development and optimization of the proposed StaP. The experimental results validated the effectiveness of the proposed StaP.**

***Keywords*—cloud environment, status prediction, heterogenous, dynamic.**

I. Introduction

The large-scale utilization of cloud computing services for hosting industrial applications has led to the emergence of cloud service reliability as an important issue for both cloud service providers and users [1]. Cloud environment employs plenty of physical machines that are interconnected together to deliver highly reliable cloud computing services. Although the crash probability of a single virtual machine might be low, it magnifys across the whole cloud environment [2].

To enhance cloud service reliability and reduce the subsequent losses, the future status of virtual machines, such as missing, locked, paused, deleting, the time taken to crash, etc., should be predicted before it happens. However, most existing methods ignore the following characteristics of actual cloud environment and this will result in bad performance of status prediction [1], [3]: (1) the cloud environment is dynamically changing, which requires response in real time;(2) cloud environment consists of many heterogeneous physical machines, which bring about different items to deal with, such as fan speed, CPU temperature, memory, disk error, application updating, software architecture, etc.

In this paper, to tackle the above problems, we propose a simple yet general machine learning model StaP to predict multiple machine status. StaP can automatically learn the representation of of different items and the correlations among them, and predict multiple statuses in real time when ready trained. The development and optimization process of the proposed StaP is detailed. The experimental results validated the effectiveness of StaP.

II. The Proposed Model

Let $V = \{\vec{v}_m \in \mathcal{R}^D | m = 1, 2, \cdots, M\}$ denote representation vectors of collected items in a D-dimension continuous space, where vector $\vec{v}_m$ represent the m-th item b_m. V is shared across all machines and can automatically be learned from collected data. Let S_n denote the collected item list of machine u_n. The length of S_n and its elements may vary with different machines. Then we denote collected data by $R = \{r_{n,m} | n = 1, \cdots, N, m \in S_n\}$, where $r_{n,m}$ stand for the collected value of u_n on b_m. We use one-hot representation to represent each status, and concatenate them all to generate representation $\vec{y}_n$ of all statuses of machine u_n [4].

Based on these denotations, we can represent the conditional probability that machine u_n takes value b_m by adding the corresponding value $r_{m,n}$ as scaling factor as follows:

$$\begin{aligned} p(\vec{v}_m|\vec{y}_n) &= (p_r(\vec{v}_m|\vec{y}_n))^{r_{m,n}} \\ &= \frac{\exp\left(r_{m,n}\vec{v}_m^T \boldsymbol{W} \vec{y}_n\right)}{\left(\sum_{m \in S_n} \exp\left(\vec{v}_m^T \boldsymbol{W} \vec{y}_n\right)\right)^{r_{m,n}}}, \end{aligned} \tag{1}$$

where $\boldsymbol{W} \in \mathcal{R}^{D \times C}$ is a parameter that need to be learned from data. And then, according to the product rule in probability theory, we can get probability that u_n takes all values of S_n as follows:

$$\begin{aligned} p(S_n|\vec{y}_n) &= \prod_{m \in S_n} p(\vec{v}_m|\vec{y}_n) \\ &= \frac{\exp\left(\left(\sum_{m \in S_n} r_{m,n}\vec{v}_m\right)^T \boldsymbol{W} \vec{y}_n\right)}{\left(\sum_{m \in S_n} \exp\left(\vec{v}_m^T \boldsymbol{W} \vec{y}_n\right)\right)^{\sum_{m \in S_n} r_{m,n}}}. \end{aligned} \tag{2}$$

Finally we get the objective function as the log likelihood over all the machines as follows:

$$\ell_{StaP} = \sum_{n=1}^{N} \log p(\vec{y}_n|S_n) - \eta||\Theta||^2, \tag{3}$$

where η is the regularization constant and Θ are the model parameters ($\Theta = \{\boldsymbol{W}, V\}$).

Training model is to find the optimal parameters Θ that can maximize Eq. 3. With elementary algebraic manipulations, we can change the training target into:

$$\Theta^* = \underset{\boldsymbol{W},V}{\operatorname{argmax}} \sum_{n=1}^{N} \left\{ \left(\sum_{m \in S_n} r_{m,n} \vec{v}_m \right)^T \boldsymbol{W} \vec{y}_n - \sum_{m \in S_n} r_{m,n} \log \sum_{m \in S_n} \exp\left(\vec{v}_m^T \boldsymbol{W} \vec{y}_n\right) \right\} - \lambda ||\Theta||^2 \tag{4}$$

As direct optimizing would suffer high computational cost, we resort to the negative sampling technique [5] for efficiency. The optimizing process is shown in Algorithm 1, where $\sigma(x)$ means the logistic function.

Algorithm 1: Learning algorithm

Input: R, $Y = \{\vec{y}_n\}$,learning rate η, maximum iterations $maxIt$, sampling number k;
Output: $\boldsymbol{W}$, V;

```
Initialize W, V, t = 0, define φ_n = Σ_{m∈S_n} r_{m,n} v_m;
while t++ < maxIt do
    for n = 1; n ≤ N; n++ do
        W = W + ησ(−φ_n^T W y_n) φ_n y_n^T;
        for m ∈ S_n do
            v_m = v_m + ησ(−φ_n^T W y_n) r_{m,n} W y_n;
        for i = 1; i ≤ k; i++ do
            sample negative sample y_i;
            W = W − ησ(φ_n^T W y_n) φ_n y_i^T;
            for m ∈ S_n do
                v_m = v_m − ησ(φ_n^T W y_n) r_{m,n} W y_i;
    Update W = W − 2ληW, V = V − 2ληV;
return W, V;
```

With the ready trained parameters $\boldsymbol{W}, V$, we can predict $\vec{y}_z$ for a new machine u_z according to

$$\begin{aligned} \vec{y}_z^* &= \underset{\vec{y} \in Y}{\operatorname{argmax}}\, p(\vec{y}|S_z) = \underset{\vec{y} \in Y}{\operatorname{argmax}}\, \tilde{p}(\vec{y}_n) p(S_n|\vec{y}_n) \\ &= \underset{\vec{y} \in Y}{\operatorname{argmax}} \left\{ \log \tilde{p}(\vec{y}) + \left(\sum_{m \in S_z} r_{m,z} \vec{v}_m \right)^T \boldsymbol{W} \vec{y} \right. \\ &\quad \left. - \left(\sum_{m \in S_z} r_{m,z} \right) \log \left(\sum_{m \in S_z} \exp\left(\vec{v}_m^T \boldsymbol{W} \vec{y}\right) \right) \right\}, \end{aligned} \tag{5}$$

where $\tilde{p}(\vec{y}_n)$ is the empirical distribution of machine status representation $\vec{y}_n$ given by the R.

As the terms $\left(\sum_{m \in S_z} r_{m,z} \vec{v}_m\right)^T \boldsymbol{W}$, $\sum_{m \in S_z} r_{m,z}$ and $\vec{v}_m^T \boldsymbol{W}$ are constant for all $\vec{y}$, the process to get $\vec{y}_z^*$ would not cost much. In Eq. 5, the empirical distribution $\tilde{p}(\vec{y})$ can be considered as the prior probability of $\vec{y}$, and $p(S_n|\vec{y}_n)$ is closely related to the likelihood function. So our model can give a Maximum A Posteriori probability estimate from the modern Bayesian Perspective, while no existing approaches in this area can make it yet [5], [4].

III. Experiment

The experimental dataset contains $210,000$ ratings expressed by $1,075$ users on $2,000$ movies. A user has individual information, such as gender, age group and his rating list. We choose this dataset because a user, his rating list and individual information can be mapped to a virtual machine, the set of collected items and its two different future statuses.

We employ POP and SNE as baseline models, and *weighted F1* and *Hamming Loss* as evaluation metrics[4]. They are commonly used in multi-task multi-class classification problem, which is similar to status prediction tasks in cloud environment.

TABLE I
Performance comparison.

Training ratio (%)	*weighted F1*			*Hamming Loss*		
	POP	SNE	**StaP**	POP	SNE	**StaP**
50	0.095	0.213	**0.278**	0.464	0.469	**0.467**
70	0.096	0.315	**0.350**	0.489	0.463	**0.458**
90	0.096	0.367	**0.379**	0.451	0.452	**0.443**

The experimental results are as shown in Table I. Clearly, the proposed StaP outperforms POP and SNE under different evaluation metrics all the time, as we set the training data ratio with 60%, 80% and 90% respectively. This result validates the assumption that the proposed StaP is a more proper model to predict the current status of virtual or physical machines by utilizing data that collected from cloud environment.

IV. Conclusion

In this paper, we address the problem of virtual machine status prediction for dynamic and heterogenous cloud environment. More specifically, we investigate the predictive power of collected data of different items from cloud environment and propose a simple yet general machine learning model StaP to automatically learn the representation of of different items and the correlations among them, and predict multiple statuses in real time. The experimental results validated the effectiveness of the proposed model.

References

[1] M. Dong, H. Li, K. Ota, L. T. Yang, and H. Zhu, "Multicloud-based evacuation services for emergency management," *Cloud Computing, IEEE*, vol. 1, no. 4, pp. 50–59, 2014.

[2] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: measurement, analysis, and implications," in *ACM SIGCOMM Computer Communication Review*, vol. 41, pp. 350–361, ACM, 2011.

[3] J. Liu, S. Wang, A. Zhou, S. Kumar, F. Yang, and R. Buyya, "Using proactive fault-tolerance approach to enhance cloud service reliability," *IEEE Transactions on Cloud Computing*, pp. 1–13, 2016.

[4] P. Wang, J. Guo, Y. Lan, J. Xu, and X. Cheng, "Your cart tells you: Inferring demographic attributes from purchase data," in *Proceedings of ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 251–260, ACM, 2016.

[5] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pp. 3111–3119, 2013.