CMPT-623/723

Distributed and Cloud Computing

(Spring 2022)

Workload Prediction using LSTM and Understanding Impact on Resource Provisioning Time

A Research Project

By:

Emran Mohannad A.Altamimi (1510662)

Submitted to

Dr. Khaled Khan

College of Engineering,

Computer Science & Engineering,

Qatar University

Abstract

The number of cloud providers who provide computing resources to users is increasing day by day. Each of them is trying to provide the best service to their users in terms of efficiency and time. They have to do this with maximum resource utilization to ensure that running costs are kept minimum at all times. The quantity of requests coming from users varies from time to time. Sometimes there may be sudden spikes during which more resources will have to be made available. Whereas, on some other occasions, there may be a sudden dip during which some resources can be freed. If we rely on systems that wait to reach a threshold workload to increase the available resources the reconfiguring time may be more than the maximum response time expected by the user. This will lead to a degradation in the Quality of Service provided to the user which may ultimately result in losing the user. Hence, it is desirable to have a workload prediction mechanism to forecast the future workload which will enable timely provisioning/shutting down of computing resources available in a cloud. In this study, we present a workload prediction mechanism that uses Long short-term memory (LSTM). We also present a simulation for comparing the execution time required for resource provisioning in a cloud environment for the predicted and actual workloads.

Keywords-Cloud Computing, Workload Prediction, LSTM, Virtual Machines, Dynamic Provisioning

Introduction

Offering computing resources (including CPU, memory, network, and virtualized servers) via the internet has become a popular business model. Cloud providers must have optimization techniques in place to deliver the computing resources as efficiently as possible to customers. These objectives include minimizing costs while maximizing revenue. Additionally, forecasting individual resource utilization becomes significant to efficiently deploy the most demanding resources. On the other hand, workload prediction provides more precise forecasting of future resource demand, making it easier to make informed decisions about how to best allocate resources in a cloud data center. Workload has several characteristics that describe how it behaves, and past workload data is one of the parameters that must be considered when forecasting workloads. These characteristics can be expressed in a variety of ways, including resource type and how much resources are consumed by the currently running workload. A workload may exhibit one or more of the following characteristics: geographic location, type of resource restriction, bandwidth on the network requirements, and requirements for security.

Cloud service providers leverage hardware virtualization to deploy several (VMs) with varying allocations of resources on a single physical machine (PM). Multiple applications are hosted on virtual machines (VMs) in a cloud. Because the demand for every Virtual machine on a physical machine fluctuates, it is possible for a PM to become overloaded, which means that the resource demand from its VMs exceeds the total available resources on the PM. A load imbalance in a PM has a detrimental effect on the effectiveness of all virtual machines (and, consequently, all applications) that run on the PM. When customer applications are given insufficient resources, it is also a violation of the Service Level Agreement (SLA). It is a contract between a client and a supplier that ensures the application's performance when the client employs the cloud service. A cloud service should mitigate PM overload to meet the SLA and ensure that virtual machines have access to the resources they require. As illustrated in Fig. 1, Based on the historical data, the forecasting system could acquire past data from the resource manager and afterward submit the projected workload for every task to that same resource manager.

Based on the predicted results, the resource manager can assign each VM to PMs (Physical Machine), which can then be verified. This is referred to as dynamic provisioning.

Cloud computing addresses the aforementioned issues by employing dynamic resource provisioning to the resource manager learned from workload patterns and characteristics such as the rate at which requests arrive and the distribution of service time. That is to say, additional resources can be allocated during high demand peaks and dispatched during low demands, maximizing the utilization rate of resources, and minimizing Cloud resource cost without compromising the quality of service (QoS) provided to end-users (Fig. 1).

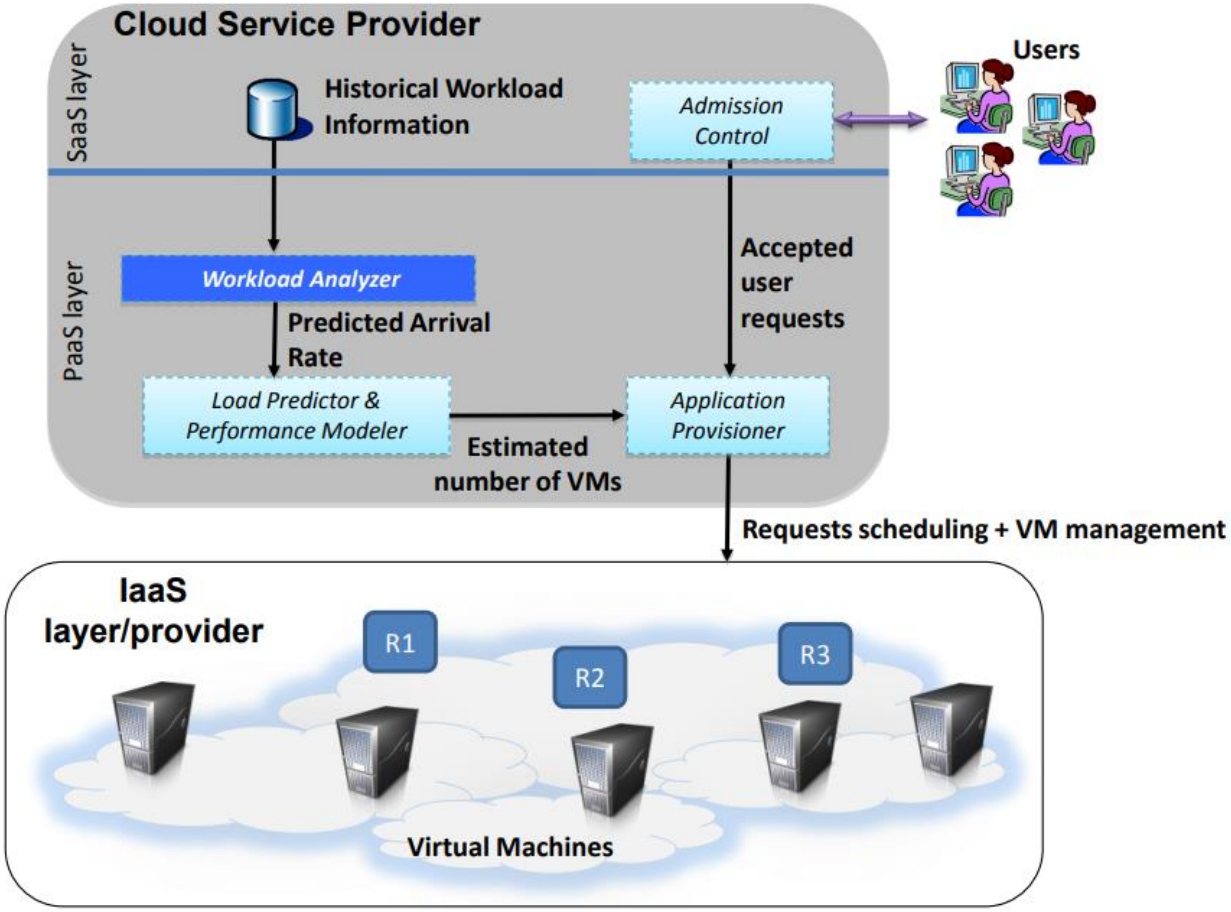

Fig. 1 Cloud Provisioning Architecture

To ensure the success of dynamic provisioning, the appropriate amount of resources must be determined for deployment in a specific duration of time attaining quality of service requirements even with fluctuating workloads, such as those encountered by Cloud applications. This issue has been addressed primarily through the use of reactive approaches, controlling resources when certain thresholds are exceeded. In proactive approaches, however, the workload is anticipated and reacted to before these workload variations manifest in the system. In the proactive approaches, techniques that monitor, predict (for example, by calculating QoS parameters ahead of time), adjust to, and acquire the correlation among QoS objectives for applications, existing allocation of Cloud resources, and alterations in workload patterns are typically used to change the configuration of resource allocation on the fly.

This study thus focuses on developing a workload prediction system using the LSTM model by using the Wikipedia workload dataset. We also used a simulation to compare the execution time required for resource provisioning for the predicted and actual workloads.

The remaining sections are organized as follows. Section II provides a discussion of the related work. Section III is the methods section where we present the system architecture and the forecasting model that we used. Section IV presents a discussion of the obtained results. Finally, in section V we present the conclusion and future work.

Literature Review

Several approaches have been used for workload prediction in different studies by researchers in this field.

Studies on workload prediction can be classified into 2 – approaches based on homeostatic prediction and approaches based on history-based prediction. The former category includes studies that use a value (e.g., mean of previous workloads) to increment or decrement the current workload as a prediction of the next workload[1]. the value used can be a constant or a variable (which is calculated based on previous workloads). The latter includes those methods which extract patterns after analyzing instances of previous workloads. These are simpler and are commonly used. They make use of time-series data to perform history-based predictions. Some common models that are used by these approaches are the Hidden Markov Model, Autoregressive Integrated Moving Average (ARIMA) model, Autoregression model, and moving average model. Such an approach was taken by the authors of [2]. They made use of a neural network and self-adaptive differential evolution for workload prediction from historic time-series data. They could achieve a considerable reduction in prediction error. Pushpalatha et al. [3] used the Chaotic Fruitfly Rider Neural Network for workload prediction from historic data. This prediction was used to achieve power optimization with the help of Virtual Machine (VM) migration and optimal switching policy.

Studies that focused on the dynamic provisioning of resources as per the workload changes can be divided into 2 categories-reactive approaches and proactive approaches. In the first category, changes in throughput and utilization are observed to respond to changes in workload. Some pre-defined thresholds are used to trigger this change. In this approach, the QoS may get impacted since more resources may get added only with a reconfiguration time which may be more than the time with which this change occurred. Such an approach was used by the authors of [4] in which they derived certain performance constraints which were used to trigger scaling up/creation of VMs. Whereas, in the other category changes to the workload are made in advance by predicting the future workload[2][3]. This approach allows reconfiguring resource allocation straight away. Such a proactive approach was used by the authors in [5]. Domain knowledge embedding regularization neural network was used to predict the workload on a large scale. Domain knowledge which was extracted from the statistical properties of a workload was embedded into an artificial neural network for linear regression for getting better workload predictions. Proactive workload prediction using the ARIMA model was done by the authors of [6]. They also studied how it impacted the QoS provided to the users by using simulations.

Several machine learning approaches have been used by different authors in their studies. Neural networks are widely preferred for workload prediction[2][3][5]. Prototype-based clustering and Density-based clustering were used to cluster similar tasks together by the authors of [7]. This clustered data

was used to train 3 machine learning models - Bayesian Ridge Regression, ARIMA, and Long Short Term Memory. They found that training using clustered data improved the prediction accuracy of these models. Support Vector Regression was used for workload prediction by the authors of [8]. This along with predicted VM performance factors were combined in this study to trigger VM migration and they found that this early triggering helped to improve resource availability. A new model called AME-WPC was used by the authors of [9] to use classification as well as regression for workload prediction. The testing was done using Random Forest. Some studies also used ensemble-based models like the one used by the authors in [10] to predict CPU load in a cloud. This model was 2-layered consisting of a predictor optimization layer and an ensemble layer. The first one could uninterruptedly incorporate new instances as well as destroy those with deficient performance. The other layer was meant to produce the final prediction depending on the results obtained from the several instances present in the first layer.

Methodology

System model and architecture

Our work is based on a system model that includes a Cloud service provider offering SaaS (software as a service) services to customers and is supported by a platform as a service layer. In turn, the layer communicates with another layer which is the infrastructure as a service provider (e.g., A third-party service provider). Web requests are received by the target SaaS provider and processed by the machines in the IaaS layer of the stack.

In order to scale up the infrastructure as needed; the targeted supplier runs a group of VMs that handle end-user requests. SaaS providers use a single virtual machine configuration that includes the CPU's power, memory, and storage capacity to simplify infrastructure management and leverage user profiling data. The provider will be aware of the VM's expected performance because, as previously stated, it profiled the application in the chosen VM configuration.

Because each VM runs a single instance of the application and because Cloud providers currently are not designed with dynamic changes to the specifications of the virtual machines with zero downtime. That is why changing how many VMs are deployed is the optimal way to leverage Infrastructures for elastic computing, it also provides increased fault tolerance and better resilience to VM break downs performance degradation (Due to the fact that the failure of one VM does not affect the others, the application may continue to serve consumers).

The applications on the web that are being targeted are the ones being discussed here. Client requests are made up of HTTP requests handled by a web server operating on the VMs. In the SLA, response time is the maximum duration to serve user requests, and rejection rate, calculated as the ratio of inbound requests that cannot be served, are two system-relevant QoS targets.

One of the most distinguishing features of Cloud computing is its elasticity, i.e., scaling up and down the infrastructure in response to application demands. On the other hand, creating new VMs is not a real-time operation. Launching a new VM can take an extended period of time, depending on the Cloud provider's infrastructure architecture and hypervisor policies. The startup time is sufficient for clients to notice, and it has a significant impact on the user experience, which may cause them to abandon the service. Financial losses associated with a decline in user numbers are only a part of the story as the software supplier could potentially also be held accountable for poor service quality (QoS).

While idle VMs may be advantageous for accommodating spikes in request volume, they tend to be highly inclined to be idle, lowering the utilization rate and raising the cost of operations. Another issue arises when the increased request volume exceeds the capacity of the standby VMs to handle the increased load. As a result, a novel approach to the Cloud provisioning problem is required. One approach under consideration is workload prediction: accurate forecasting of future service requests from end-users enables targets for SLA's Quality of Service to be satisfied while utilizing fewer Cloud resources than would be necessary otherwise.

We have developed a workload prediction mechanism that is implemented in an adaptive provisioning mechanism for Cloud environments. To understand how the prediction module plays a part in the adaptive provisioning mechanism this section will explain the mechanism which consists of three critical components. These components are intended to mitigate the uncertainty inherent in workload patterns and to minimize forecasting error while maintaining optimal system utilization.

The components are:

- Application Provisioner: The application provisioner inspects the admission control module requests and assigns VMs to handle them. Additionally, it monitors the VMs' performance. This data is communicated to the next component explained in the next bullet point. Furthermore, this module gives the anticipated number of VMs needed by the program to the application provisioner. Providing additional VMs or decommissioning unused VMs if the anticipated number of VMs is larger or lower than the supplied number.
- Load forecasting and Performance Modeler: Determines how many VMs to allocate according to the Workload Analyzer module's predicted demand and the Application Provisioner's assessed effectiveness of operating VMs. The performance is modeled using queueing networks, and it returns the smallest number of VMs capable of meeting the QoS metrics derived from the request rate of arrival forecast.
- Workload Analyzer: Estimates the application's future demand. This information is then passed to the module that contains the load forecast and performance modeler.

At the start, previous data is fed to the LSTM model (i.e., the workload analyzer). While the system is functioning, it provides a one-time estimate of the workload. The duration of the time period could be customized for the application’s requirements. The condition for optimal utilization rate is to give enough time for the deployment of additional VMs. As a result, time frames as little as 10 minutes are considered appropriate, of course, this comes down to the cloud provider in question.

Dataset

In our work, a publicly available dataset from Wikipedia workload was forecasted. The total workload for the entire year 2021 was retrieved. There are three types of information available via the API (Application Programming Interface) for the aggregated workload. The first information is the project type, there are 13 projects in total, with each project spanning over a hundred languages, the API only provides the data for a specific project in a specific language. The second information available is the access type, the API can retrieve and distinguish data of several access methods (desktop, mobile application, and mobile web). The third type of information is the agents; users, spiders, or automated.

We retrieved three ‘datasets’ to test our method, for each type of information. We designed a python API that downloads and stores the data for each information type. For the projects, we have selected

the most common 15 languages and downloaded 75 datasets for the individual projects. For the access and agent types, each had three data sets for each category. Figure shows an example of the datasets of each category. Afterward, we aggregated the datasets for each category to serve as an evaluation baseline for our new proposed method. Fig. 2 shows the aggregated data for each category.

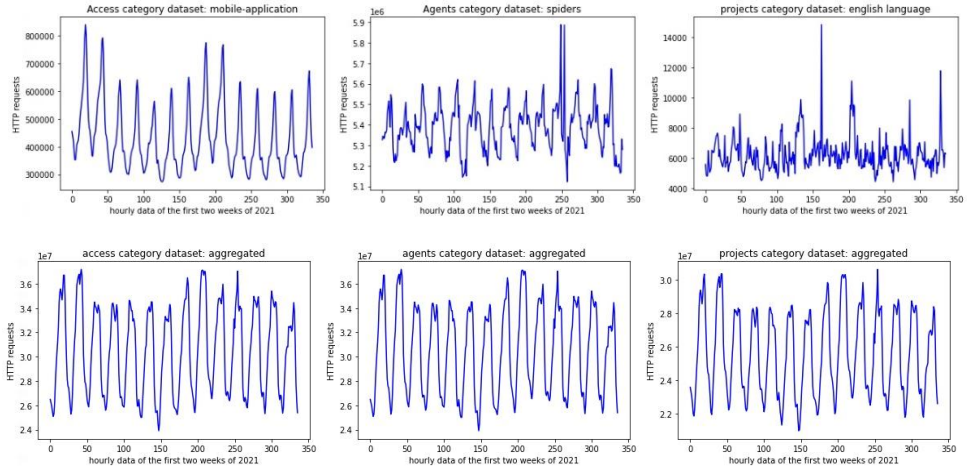

Fig. 2 Category wise aggregated data

The forecasting model

Long short-term memory or LSTMs have been used in Recurrent Neural Networks (RNN) to acquire the ability to make predictions from variable-length sequences. It can work on any type of sequential data not only time-series data unlike other ML models like ARIMA. An LSTM unit consists of a cell, input gate, output gate, and forget gate as seen in Fig. 2. The cell stores information for long periods by updating the internal state. The data going in and out of the cell in controlled by the three gates of the cell. The primary goal of each LSTM cell is to memorize significant segments of the sequence observed up to this point and to discard the arbitrary segments. Developing an efficient and optimum LSTM-based model can be challenging and requires meticulous hyperparameter tuning.

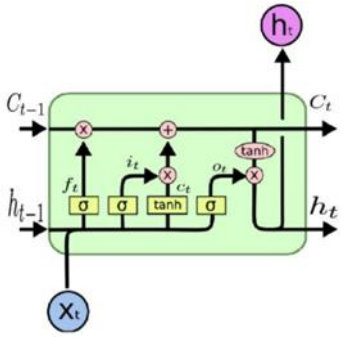

Fig. 2. LSTM cell

Recurrent neural networks built on LSTMs are arguably the most powerful technique for training on sequential datasets, with time series being an example. The true capacity of LSTM is realized when training on large datasets with intricate patterns. In contrast to ARIMA, they do not make specific prior

assumptions, like time-series data stationarity or the presence of a Date field. One of the drawbacks is that LSTM is hard to analyze making developing an intuition for their behavior difficult. Additionally, meticulous hyperparameters optimizations are required to obtain acceptable results.

Each model built has the same architecture, which consists of 80 neurons or cells in the input layer, a fully connected hidden layer with 40 cells (a higher number of cells will be prone to overfitting while a lower number will not be able to capture complex patterns from the dataset), and an output layer. The mean squared error was selected as the loss function and the Adam optimizer was used. The data was trained on 90 epochs.

The input data included the past 24 hours' workload and the following features for each instance:

1. The hour of the day
2. The day of the month
3. The month of the year
4. A weekday or a weekend

Execution Time Estimation through Simulation

In order to observe how the accuracy of workload prediction impacts the QoS provided to users, we performed a simulation to compare the execution time required for resource provisioning for the predicted and actual workloads. This execution time refers to the time taken by the broker to provide the required computing resources to a user when a user request comes in. It is an important metric that is used to measure the QoS provided to the users. Most cloud providers will include a maximum limit for this execution time in their SLAs. If this time is exceeded some users may switch to other cloud providers hence, it is of high importance.

The simulation was done with the help of the CloudSim[11] toolkit. The architecture of the simulated data center is similar to what was described in Section III. It has 1000 hosts which host 300 VMs. Each of these hosts has 8 cores and 16GB of RAM. Each VM had 1 CPU core,2GB RAM, and 10GB storage. The applications running on these VMs are scheduled in a time-shared manner. In the beginning, the experiments were conducted with 50 VMs. The number of requests coming in during each hour had to be assumed to be coming in batches of 1000 in order to reduce the running time of the simulation. Even after making this assumption, the simulation for each hour required approximately 500s to execute. So, if we try to simulate a month's data, we would require 100 hours to run the simulation with the predicted data and another 100 hours to run the simulation with the actual data (assuming that there are 30 days in that month). Hence, we decided to run the simulations for the 24 hours of 1st December 2021 and closely observe the difference in actual vs predicted execution times for resource allocation.

The simulation experiment also provided a platform to observe how effective resource utilization can be done according to the predicted workload. For this purpose, we conducted experiments by varying the number of VMs that are available and observed the changes in execution time required for resource provisioning. These experiments included the simulation for the requests that came in the 1st hour of 1st December 2021 with the availability of 50 VMs,300 VMs and 1000VMs. We also conducted a simple experiment with 2 datacenters each having 2 hosts,20 VMs and 40 user requests(cloudlets).This experiment was conducted to observe how the available VMs were used by the broker for resource provisioning.

Discussion

Prediction Results

Our method works by building a model for each dataset in each category and aggregating the forecasted workload of all the models. Fig. 3 shows the predicted and real workload values for each dataset in the access category along with the predicted and real workload values in the aggregated dataset as a benchmark.

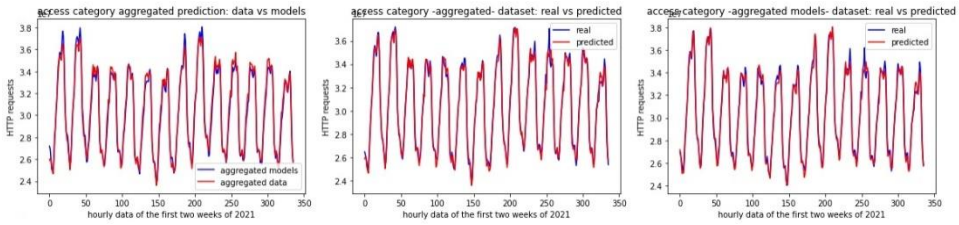

Fig. 3. Predicted vs Real Workload

Our results have shown improvements when aggregating the forecasted workloads. Table I shows the results. The evaluation metrics used are the MAE(mean absolute error), the RMSE (root mean squared error), and the NRMSE (normalized root mean squared error). The scales of the workload predictions are huge (in the order of 10^7) that is why the NRMSE gives a better intuition of the accuracy of the workload, with the errors being around 2 percent.

Category:	Projects		Access		Agents	
Aggregate/ individual	Aggregate	Individual	Aggregate	Individual	Aggregate	Individual
MAE	659556	500448	1000759	804721	865369	581941
RMSE	877231	722765	1271562	1025327	1100925	800481
NRMSE	0.0364786	0.03205892	0.04379774	0.0364236	0.03792028	0.02978689

Table I: Results of Workload Prediction

The results show a clear improvement when taking the prediction of the individual datasets over the entire dataset, meaning aggregating workload definitely causes some information loss and obscure valuable patterns to be discovered by the model.

Taking a closer look at the results, we have seen a significant improvement in the agents' category even though only three forecasting models were taken. We reason that because the agents' category contains two distinct request types (spiders and automated) and users their pattern and behaviors differ dramatically. When we inspected the training errors in the agents' category, we found that the NMRE was close to 1% in both the spiders and automated agents, and 2.5% of the agents' users. This indicates that automated agents are more systematic and predictable.

In the projects category, we attribute the improvement to two factors, first is the wisdom of the crowd, since there are 75 different models all predicting the load, the effect of some overshooting by some models will be cancelled by undershooting in others, this effect is also utilized in other well-known ML

algorithms also known as bagging. While bagging builds numerus models from the same dataset and averages the prediction.

The other effect is by taking several languages and projects, we reason that some patterns about the behavior of each country will be learnt from the models, for example different weekends days and different school hours due to different time zones.

To our surprise, we also found an improvement in the access category, we are not sure what is the improvements attributed to, but we reason that mobile users are usually more scattered and less predictable than desktop users who are more inclined to use their devices in more predictable hours.

Simulation Results

The simulation experiment was conducted for the actual and predicted values for each hour on 1st December 2021. The execution time required for resource provisioning for the predicted and actual workloads was found in this experiment. The values obtained have been shown in Fig. 4. It can be seen that the execution time for resource provisioning from the predicted workload simulation is very close to that obtained from the actual workload simulation.

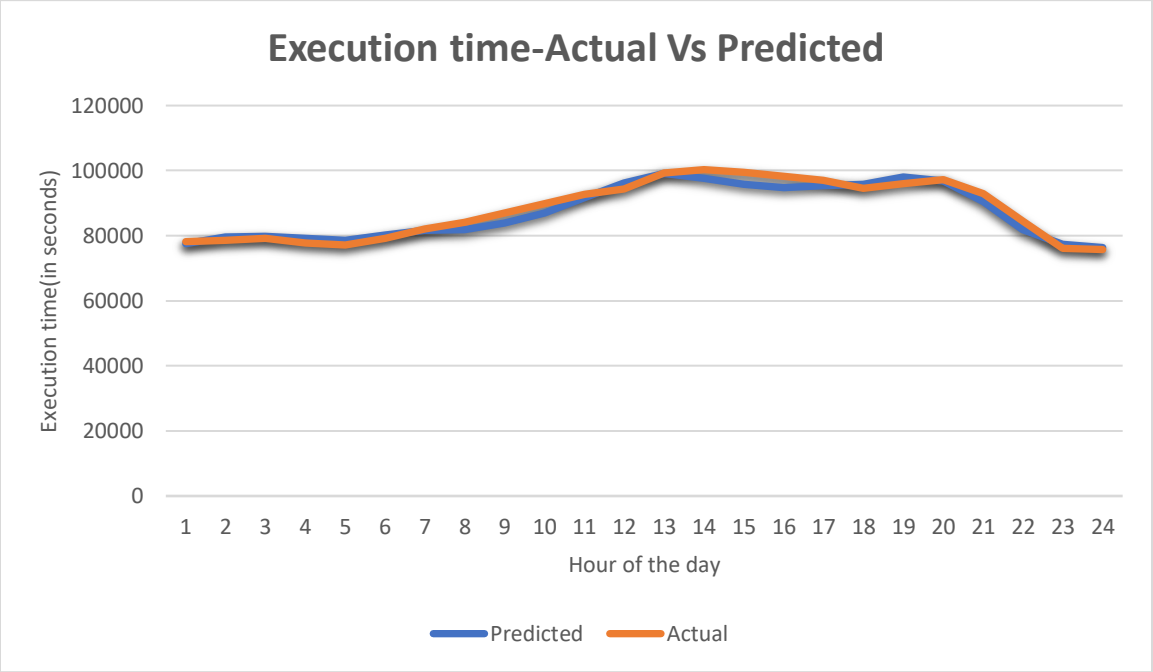

Fig. 4. Predicted vs Real Execution Time

Table II shows the execution time for resource provisioning with different available number of VMs for the requests that came in the 1st hour of 1st December 2021. It can be observed that the execution time for resource provisioning decreases with increasing availability of VMs.

Number of available VMs	Execution time (in seconds)
50	465097.83
300	77525.421

1000	23265.889

Table II: Execution time with varied VM availability

Finally, in our last experiment that we conducted to observe the actual VM utilization by cloudlets, we could observe that only 11 VMs were required to serve 40 user requests or cloudlets, even though 20 VMs were available. So, such simulations can help to gather insights to make decisions to scale up/scale down the number of VMs according to the predicted workload.

Conclusion, challenges and future direction

In this work, we designed a workload prediction model based on an LSTM model. The prediction model is used to help in dynamic provisioning of workload to improve the QoS in the cloud. We built several models to harvest the power of the wisdom of the crowd to achieve superior performance. Then the impact on QoS was evaluated on the CloudSim toolkit.

Workload prediction has huge impact for provisioning, however it is currently limited to only short term predictions as discussed in this work, the reason is that short term predictions have superior performance and are much more predictable than long term predictions, however it limits the applications workload predictions has to offer. This problem is much more severe for large cloud providers who can not afford to have a poor of QoS due to poor forecasting performance.

A future direction is to utilize models with confidence intervals and dynamically incorporate confidence in the application provisioner. For example, with forecasting with low confidence intervals, we add provision more VMs to the applications as a factor of safety, and with higher confidence intervals we trust the prediction model more and only add the necessary VMs. An optimization method is a future research direction. To the best of our knowledge, no work has been done in this area.

Bibliography

[1] L. Yang, I. Foster, and J. M. Schopf, “Homeostatic and tendency-based CPU load predictions,” *Proc. - Int. Parallel Distrib. Process. Symp. IPDPS 2003*, p. 9, 2003, doi: 10.1109/IPDPS.2003.1213129.

[2] J. Kumar and A. K. Singh, “Workload prediction in cloud using artificial neural network and adaptive differential evolution,” *Futur. Gener. Comput. Syst.*, vol. 81, pp. 41–52, Apr. 2018, doi: 10.1016/J.FUTURE.2017.10.047.

[3] R. Pushpalatha and B. Ramesh, “Workload Prediction Based Virtual Machine Migration and Optimal Switching Strategy for Cloud Power Management,” *Wirel. Pers. Commun.*, vol. 123, no. 1, pp. 761–784, Mar. 2022, doi: 10.1007/S11277-021-09157-W/TABLES/3.

[4] N. Bonvin, T. G. Papaioannou, and K. Aberer, “Autonomic SLA-driven provisioning for cloud applications,” *Proc. - 11th IEEE/ACM Int. Symp. Clust. Cloud Grid Comput. CCGrid 2011*, pp. 434–443, 2011, doi: 10.1109/CCGRID.2011.24.

[5] L. Li, M. Feng, L. Jin, S. Chen, L. Ma, and J. Gao, “Domain Knowledge Embedding Regularization Neural Networks for Workload Prediction and Analysis in Cloud Computing,” *https://services.igi-*

global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-7998-5339-8.ch055, pp. 1158–1176, Jan. 1AD, doi: 10.4018/978-1-7998-5339-8.CH055.

[6] R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, "Workload prediction using ARIMA model and its impact on cloud applications' QoS," *IEEE Trans. Cloud Comput.*, vol. 3, no. 4, pp. 449–458, 2015, doi: 10.1109/TCC.2014.2350475.

[7] J. Gao, H. Wang, and H. Shen, "Machine Learning Based Workload Prediction in Cloud Computing," *Proc. - Int. Conf. Comput. Commun. Networks, ICCCN*, vol. 2020-August, Aug. 2020, doi: 10.1109/ICCCN49398.2020.9209730.

[8] B. R. Raghunath and B. Annappa, "Virtual Machine Migration Triggering using Application Workload Prediction," *Procedia Comput. Sci.*, vol. 54, pp. 167–176, Jan. 2015, doi: 10.1016/J.PROCS.2015.06.019.

[9] K. Cetinski and M. B. Juric, "AME-WPC: Advanced model for efficient workload prediction in the cloud," *J. Netw. Comput. Appl.*, vol. 55, pp. 191–201, Sep. 2015, doi: 10.1016/J.JNCA.2015.06.001.

[10] J. Cao, J. Fu, M. Li, and J. Chen, "CPU load prediction for cloud environment based on a dynamic ensemble model," *Softw. Pract. Exp.*, vol. 44, no. 7, pp. 793–804, Jul. 2014, doi: 10.1002/SPE.2231.

[11] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw. Pract. Exp.*, vol. 41, no. 1, pp. 23–50, Jan. 2011, doi: 10.1002/SPE.995.