A preliminary study of machine learning workload prediction techniques for cloud applications

Dionatrã F. Kirchoff, Miguel Xavier, Juliana Mastella and César A. F De Rose
Polytechnic school
Pontifical Catholic University of Rio Grande do Sul
Av. Ipiranga 6681 – 90.619-090 – Porto Alegre – RS – Brasil
Email: dionatra.kirchoff@acad.pucrs.br

***Abstract*—Cloud computing has transformed the means of computing in recent years with several benefits over traditional systems, like scalability and high availability. However, there are still some opportunities, especially in the area of resource provisioning and scaling [13]. Since workload may fluctuate a lot in certain environments, over-provisioning is a common practice to avoid abrupt Quality of Service (QoS) drops that may result in Service Level Agreement (SLA) violations, but at the price of an increase in provisioning costs and energy consumption. Workload prediction is one of the strategies by which efficiency and operational cost of a cloud can be improved [13]. Knowing demand in advance allows the previous allocation of sufficient resources to maintain QoS and avoid SLA violations [1]. This paper presents the advantages and disadvantages of three workload prediction techniques when applied in the context of cloud computing. Our preliminary results compare ARIMA, MLP, and GRU under different cloud configurations to help administrators choose the more appropriate and efficient predictive model for their specific problem.**

***Index Terms*—workload prediction, cloud computing, resource efficiency**

I. Introduction

Cloud computing has evolved at a rapid pace and has become popular in today's enterprise. The adoption of cloud computing has been motivated by the prominent scalability, availability and the promise for lower infrastructure costs [13]. Large-scale component-based enterprise applications that leverage cloud resources expect Quality of Service (QoS) guarantees in accordance with Service Level Agreements (SLA) between the customer and service providers. In the context of cloud computing, auto-scaling strategies promise to ensure the QoS properties to applications, while making efficient use of resources. Despite the perceived advantages of auto-scaling, getting the full potential of auto-scaling is difficult due to several challenges arising from the need to accurately estimate the use of resources due to the unpredictable variation of customer's workload patterns. Many papers explored strategies to better adapt environments to applications using learning algorithms from experiences [13] [17] [20] [24]. They stated that predictive models could be applied over workload behaviors to accurately allocate the resources that are necessary to satisfy QoS policies and SLA [1].

Workload prediction algorithms belong to a regression problem, which means that variables are estimated over time [8]. Thus, the learning phase in a Machine Learning algorithm does not require a robust infrastructure to run as opposed to classification problems, which need high-priced infrastructure to compute a massive collection of data. Amiri et al. [1] have surveyed studies related to workload prediction in clouds. The works account for similar problems in complexity, but they are unware of performance metrics and model's accuracy, which are essentials for QoS and SLA maintanance.

This work presents a trade-off analysis among the state-of-the-art workload prediction techniques for cloud applications. To compare the algorithms we implemented the most popular Machine Learning-driven models that are broadly adopted in recent studies [1]: ARIMA, MLP and GRU.

II. Background

A. Workload prediction characterization

In computer science, the workload is the amount of work performed by a computer in a given period demanded for some application. The collection of this information makes possible to map the application's behavior and apply prediction techniques to discover future behaviors and forecast infrastructure demands. In the cloud computing, the application prediction is an essential step for the efficient resources management [1].

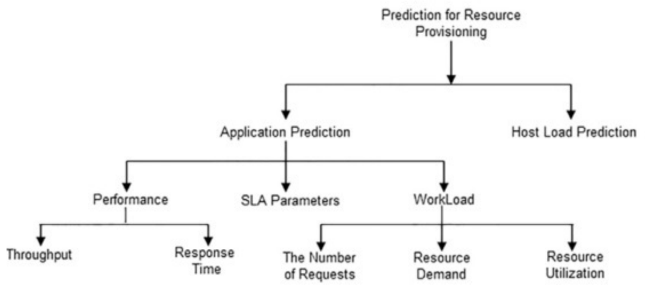

Fig. 1. Different levels of predictions by Amiri and Mohammad-Khanli [1]

Amiri and Mohammad-Khanli [1] researched the concept of 'workload' and they found that it is interpreted in different ways in the literature as it is shown in Figure 1.
In some papers such as Liang et al. [15] and Kumar et al. [13], they consider the application's workload is equivalent to the number of requests for the application. In this approach, the future number of requests is the output of the prediction method.
The case of Jiang et al. [12] the workload is interpreted as the future demand of Virtual Machines (VMs). For Garg et al. [7]

2377-5750/19/$31.00 ©2019 IEEE
DOI 10.1109/PDP.2019.00039

the resource utilization of VMs are considered as workload. They assumed that each application is capsulated inside one VM.

In the application performance prediction, Manvi and Krishna Shyam [16] mentioned the values of throughput and response time are predicted based on the resource allocated for it. In this situation, the primary goal is to determine the best resource allocation to obtain the desired performance level.

Leitner et al. [14] predict SLA parameters of applications. They usually predict SLA violation of the application according to its workload or the resources allocated for it. In other words, data which refer to both typical QoS data (e.g., response times, availability, system load) and process instance data (e.g., customer identifiers, ordered products) could be used for prediction. Finally, host load prediction is used by Yang et al. [25] with the focus to improve resource utilization in the cloud.

This section presented different workload characterization in different literatures. The next session describes some cloud applications characteristics.

B. Cloud computing and Web applications

Singh and Chana [21] assume that cloud computing supports different types of applications. The applications may be indivisible with multiple workflows applications, such as map reduce applications, adaptive data stream, scientific applications, homogenous and heterogeneous scientific workflows, elastic and scalable applications, data intensive, network intensive and computation intensive, applications with multi-tier workloads.

The architecture of some cloud applications such as Web sites, E-commerce, are mostly designed with multiple tiers for flexibility and software reusability, these development approach has been remodeled to support the scalability demanded [10].

For Nanda et al. [19] modern web applications usually have three major components, a web server, an application server, and a database server. They are usually connected in a three-tier architecture so that resources can be allocated to each tier independently.

The demands from users for these services can vary widely based on factors such as the time-of-day and unexpected events that can trigger crowds, for example, some promotions even holidays.

Huang et al. [10] mentions, that capacity planning is a classic method to determine the number of resources for the given QoS requirement. However, the authors also explain that capacity planning is a long-term and almost static decision, and the resources are determined by the maximum Web application request rate in the target period to avoid excessive penalty. The maximum application request rate can be estimated according to a prediction model or historical data [10].

As mentioned by Kumar et al. [13] the workload prediction is one of the possibilities by which the efficiency and operational cost of a cloud can be improved. Therefore, the following section describes some approaches for application prediction.

III. Workload-agnostic Prediction Taxonomy

There are several research efforts to discover approaches to find the application's behavior. Based on this Amiri and Mohammad-Khanli [1] proposed a general taxonomy for models of the application prediction. According to the authors, prediction models could be divided into four groups. These groups include table-driven methods, control theory, queuing theory and machine learning techniques as it is shown in Figure 2.

A. Table-driven methods

In table-driven methods, the application behavior is recorded in a table for different values of the workload intensity and different amounts of resources allocated to it [1]. Interpolation is a method of constructing new data points, and it is used to obtain values not recorded in the table. This approach has some limitations, like low scalability due to the number of applications, different states of the resources allocation and different types of workloads [3]. The table building is time-consuming and demands several experiments to fill the table. So this method is considered obsolete as mentioned by Amiri and Mohammad-Khanli [1].

B. Control-theory

In control models, the goal is to control resources shared between cloud applications [1]. The control theory has a mechanism for dealing with unpredictable changes, and disturbances in systems using feedback [26].

This kind of systems can be divided into two groups, open loop control and closed group control systems [1]: In open loop control systems, the performance of the application (output) depends on allocated resources (the input signal) and the output does not affect the input to control the resources allocated to the application. In closed-loop control systems, there is a feedback loop that compares the application performance with the desirable performance. So, the controller adjusts resources allocated according to its performance goal.

A standard closed-loop control based on Zhu et al. [26], refer to the system being controlled as the target system, which has a set of metrics of interest (referred to as measured output) and a set of control knobs (referred to as control input).

Furthermore, as mentioned by Zhu et al. [26] there are some limitations to this approach such as the nonlinearity interrelationships in computing systems which makes the modeling harder to understand and to apply in practice. Also, imposes a limitation on how fast workloads or system behavior can change. There are problems with the time granularity, making it impossible to design controllers that respond to changes at shorter time scales as required in modern systems.

C. Queuing theory

The Queuing Network (QN) model can be used to predict the performance of the application and modeling the relationship between the workload and the performance criteria [1]. In the QN each server is allocated to the application as a queuing system as mentioned by Urgaonkar et al. [23], which investigate this approach with multitier Internet applications.

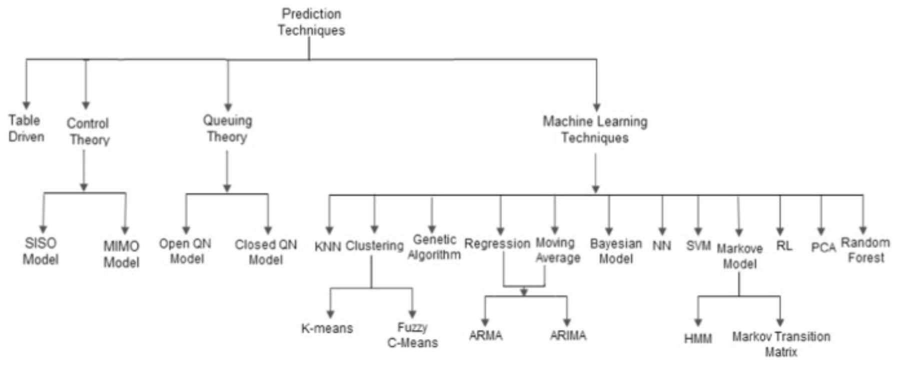

Fig. 2. The taxonomy of prediction methods based on Amiri and Mohammad-Khanli [1]

The jobs go from one queue and arrive at another queue. These models have parameters such as the requests arrival rate and the average resources requirements of requests that should be specified [23].

The open QN the jobs could be external. Thus, the number of jobs in the system varies with time [9]. In the closed QN, there is a constant population in the network and no external sources. There is a few related works in this approach for workload prediction. It is possible that the method was replaced for more straightforward implementation. In the research of Amiri and Mohammad-Khanli [1] there is not much information about new researches in this scenario.

D. Machine learning techniques

The recently proposed approaches are based on Machine Learning (ML) techniques. The ML based methods predict the application behavior in different dimensions. They are not only used to predict the future behavior of resources [12] but also are used to predict SLA Violation [14], the application performance, resources allocation and the execution time of jobs [1].

There are some problems which cannot be achieved through specific algorithms. However, if there is historical data available, it is possible to predict or derive tasks using ML techniques [8]. In other words, based on the application behavior over time as input data, can be build ML models that output a prediction for the future application behavior [1].

Considering this context, this section outlines some selected ML approaches for application forecasting, including a summary of its theoretical base, specific constraints and requirements as well. The challenges for accurate predictions are interaction with varying number of clients and high non-linearity in workload. So, the ML approach is widely being adopted for establishing more accurate prediction models, motivated by the power of working with non-linearity workloads, consequently promoting promising results [13]. Therefore, based on the above statements we chose the ML techniques to explore in our work.

IV. Machine Learning-centric Classification

This session depict three well known machine learning techniques that showed promising results in related work under similar scenarios as the one explored in this paper [13], [17], [20], [24]. They will be compared to each other in our preliminary experiments in Section V-C.

The first method is called ARIMA, which gives non-stationary time series prediction, which stands for Auto Regressive Integrated Moving Average and has successfully been applied to many fields, such as finance [3].

The second was inspired by the structure of biological neural networks [8] called artificial neural networks (ANN). The ANN or MLP(Multi Layer Perceptron) has a feedforward behavior, so it forms a directed acyclic graph. According to Russel and Norvig [22], neural networks are composed of nodes or units connected by directed links, the properties of the network are determined by its topology and the properties of the neurons.

The last approach RNN (Recurrent Neural Network) has the same building process as ANN. Nonetheless, the RNNs has connections between units form a directed cycle, so they can remember important things about the input they received, which enables them to be precise in predicting what is coming next, it is usually adopted for spoken language understanding, natural language processing (NLP), [5], [2]. In our experiments, we implemented the model GRU (Gated Recurrent unit) proposed by Cho et al. [5].

V. Preliminary Evaluation

In our preliminary evaluation, we compare three ML workload prediction techniques from the literature that are obtaining the more promising results in similar problems, namely ARIMA, MLP, and GRU, under different cloud configurations.

A. Experimental Environment

The experiments were performed on a machine equipped with an Intel Core I7-6500U processor with 2.50 GHz clock speed with 16 GB of memory. We used Python 2.7 and an interactively environment to implement and execute all techniques used in this paper (Google Colaboratory). The scripts, source code and workload traces used in our experiments are available in https://github.com/dionatrafk/workload_prediction for reproducibility purposes.

To define the benchmark used in our experiments we analyzed the characteristics of the benchmarks that are used in related work, such as [13] and choose the NASA HTTP traces, obtained from [11]. NASA HTTP contains two separated two-month long traces containing all HTTP requests to the NASA Kennedy Space Center WWW server in Florida. The traces are stored in ASCII files with one line per request. To prepare the data for the different intervals, we used the sum operation. We added the number of requests received in every time interval of interest.

According to Diaz et al. [6] a major challenge in designing ML systems is to determine the best structure and parameters for the network given the data for the ML problem at hand. These hyperparameters are chosen based on heuristic rules and are manually fine-tuned to accelerate the discovery of configurations yielding high accuracy, which is a very time-consuming process. Therefore, in our preliminary evaluation, MLP and GRU algorithms were implemented with their standard parameters and not fine-tuned to each executed dataset. In order to compare techniques, the executions were performed splitting the dataset, using about 60% for the training phase, and the other 40% used for the test phase.

For the three parameters required by the ARIMA technique $ARIMA(p,d,q)$, we applied a methodology called Hyperparameter Tuning using grid search [18] that estimates the best configuration by iterative trial and error from reviewing diagnostics. This methodology executes the ARIMA model varying the parameters in a range of 0 to 10 and gives us the best Hyperparameters for a dataset and time interval. Table I presents the used parameters for each of the evaluated configurations.

TABLE I
HYPERPARAMETERS USED IN EXPERIMENTS

Interval(min)	ARIMA(p,d,q)	MLP and GRU	
1	(0, 1, 2)	Batch size:	140
5	(2, 0, 1)	Epoch:	150
10	(0, 1, 1)	Input layer:	32
15	(0, 2, 1)	Hidden layer:	16
20	(1, 0, 2)	Output layer:	1
25	(4, 0, 1)	Dropout:	20%
30	(2, 0, 2)		
35	(4, 0, 1)		
40	(8, 0, 1)		
45	(2, 0, 2)		
50	(2, 0, 1)		
55	(10, 0, 2)		
60	(10, 0, 2)		

B. Accuracy Metrics

The following metrics measure the quality of an estimator. They are always non-negative, and values closer to zero are better [20].

1) Mean squared error (MSE): To compute the MSE, we first take the square of the difference between the actual and predicted values of every record. We then take the average value of these squared errors. If the predicted value of the i^{th} record is P_i and the actual value is A_i, divided by n which correspond a number of observations, then the MSE is the equation 1:

$$MSE = \frac{\sum_{i=1}^{n} (P_i - A_i)^2}{n} \quad (1)$$

It is also common to use the square root of the MSE called Root Mean Square Error (RMSE) [20].

C. A comparison among the prediction techniques

Table II shows prediction accuracy results obtained from executions with different time intervals for each technique. Since a smaller RMSE value means a better overall prediction, we can clearly observe that accuracy tend to get worse with increasing intervals. Its also noticeable that ARIMA obtained slightly better results when compared to MLP and GRU in this experiment.

TABLE II
RMSE PREDICTION ACCURACY

Interval(min)	MLP	GRU	ARIMA
1	16,26	16,51	14,27
5	54,34	53,04	47,86
10	96,6	94,26	86,98
15	136,36	132	135,08
20	181,07	172,3	168,39
25	207,65	199,19	191,03
30	249,55	242,86	237,73
35	280,32	271,64	263,43
40	339,86	323,98	316,6
45	390,47	368,25	360,48
50	418,97	403,8	380,1
55	493,72	465,63	439,89
60	544,08	521,66	493,90

An explanation for ARIMA results is that, as explained earlier, the algorithm was tuned with hyperparameters for better configurations in every dataset. MLP and GRU techniques were performed with fixed configurations in all of the execution tests as shown in table I and still have the possibility of improvement by further adjusting the hyperparameters. Nevertheless, both prove a high capacity for generalization and adaptation in the workloads at different time intervals, as we can see in the detailed plots, being more flexible to apply in different contexts and having a shorter execution time.

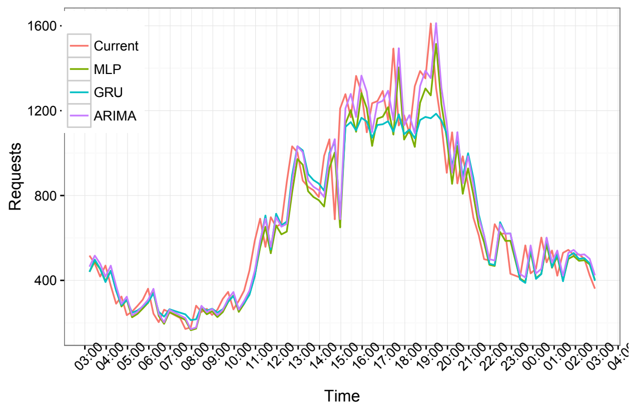

Fig. 3. Predictions with 15-minute (Zoom in) time intervals

The line graphs show a more detailed view of the predictions for 15 (Figure 3) and 60 minutes (Figure 4) confirming that all three techniques coped with the no-linearity of our test workload.

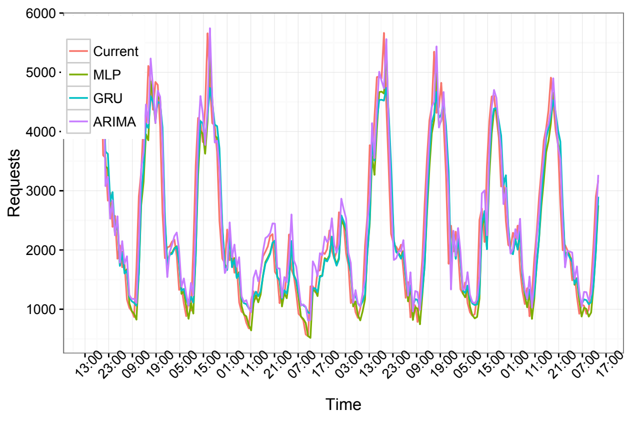

Fig. 4. Predictions with 60-minute time intervals

During the execution of the experiments, we found differences in the execution time between the techniques. MLP and GRU performed the compilation, training and test phase in seconds, regardless of the time interval of the dataset. In the other hand, ARIMA needs to be updated every time a prediction is made, since the current values need to be included in the time series to provide the forecast for the next time interval. Thus, depending on the prediction functionality needed, the execution time can represent a significant overhead the resource management strategy.

ARIMA showed to be more dependent on the number of samples than MLP and GRU. GRU demonstrated better results among neural networks, although in the execution of experiments, was observed that has a relation between the number of training samples and the *batch size* parameter, then, for good performance, it would be interesting have enough samples that are compatible with the size of the *batch size* defined in the model. ARIMA also showed to be more dependent on the number of samples than MLP and GRU, that need fewer samples to provide accurate forecasts.

During the executions, the MLP model has not suffered direct influence between the number of samples and the *batch size* as it does with the GRU. It was noticed that MLP and GRU need fewer samples to provide accurate forecasts.

Based on our preliminary experiments and the above findings concerning flexibility of use, execution time, sensitivity to the number of samples and accuracy, we recommend the utilization of GRU in scenarios similar to the one evaluated in this work.

VI. Related Works

In this section, we analyze the current state-of-the-art related to workload prediction. As mentioned in section II-A there are already several research efforts for other problem domains, tailored for their specific workload characteristics. Some of them make use of simple statistical predictors, others are already applying machine learning techniques and improving algorithms to train them for their specific problem. In the next paragraphs, we present the more relevant contributions that have similarities to what we did in this paper for the prediction of cloud computing applications.

Jitendra and Singh present a workload prediction in the cloud using an artificial neural network (ANN) and an adaptive differential evolution [13]. Their ML technique is trained with HTTP Requests using an evolutionary approach to minimize the effect of the initial solution choice. Evolutionary algorithms explore the space in multiple directions using a set of solutions [13]. The ML techniques learn and extract patterns from the workload, which can be further utilized to improving resource scaling decisions. Their implementation resulted in better results when compared with traditional ML approaches such as the training with back propagation.

Messias et al. [17] build a model that combines time series prediction models with a genetic algorithm to autoscale Web applications hosted in the cloud infrastructure. The forecasts are based on five models: naive model (Naive); autoregressive model (AR); autoregressive-moving-average model (ARMA); autoregressive integrated moving average model (ARIMA); and extended exponential smoothing model (ETS) [17]. Because none of these methods achieves the best result in all cases, the authors use a genetic algorithm (GA) to combine the benefits of each individual forecasting method and increase forecast accuracy. To evaluate the proposal they used three logs extracted from real web servers.

Prevost et al. [20] introduces a novel framework that combines workload prediction and stochastic state transition models. The goal of this work is to obtain an optimal resource allocation in the cloud by minimizing the energy consumed while maintaining the required performance levels. They used ANN and AR to forecast the application workload in cloud data centers. The authors show that both models can adequately predict future workloads, but the relative RMSE values for the Linear AR predictor obtained more accurate results then ANN. Although they compare a wide range of time intervals, varying from 1s to 90s, they are all relatively short, and longer intervals would probably favor ANN, that can better handle non-linear workloads.

Calheiros et al. [4] presents the implementation of a cloud workload prediction module for Software as a service (SaaS) providers and its impact on cloud applications QoS. The predicted load is used to dynamically provision VMs in an elastic cloud environment. Their prediction technique is based on the ARIMA model. The system was evaluated with real traces of requests to the web servers from the Wikimedia Foundation. The authors justified this choice based on their experiments, were ARIMA had better accuracy results than the more complex ANN models, because the used workloads were more linear. To evaluate the impact of accuracy concerning efficiency in resource utilization and QoS of user requests, the experiments were performed via simulation using the CloudSim toolkit, which contains a discrete event simulator and classes that enable users to model cloud environments. In their workload context, the simulation achieves an average accuracy of 91 percent, which results in minimal impact on the QoS.

Yang et al. [24] proposed an auto-scaling mechanism for virtual resources at different levels in service clouds. Since the workload trend in this case is linear for a relatively short period, they need a method which can quickly adjust its model to cope with workload variations. An LR prediction model was proposed and compared to two versions of ARMA: Mean (The predicted workload is the mean workload over the workloads in the window) and Max (The predicted workload is the maximum workload over the workloads in the window). LR obtained better results in their scenario. Because none of the techniques was better for all configurations, authors recommend that the length of the sliding window should be used to choose the best technique for a specific situation. In their experiments, the authors conclude that their approach reduces the final cost of the resources and generates less SLA violation.

Although all the works motioned above showed very good results for the scenarios they were tuned for, we were interested to verify the efficiency of the above core prediction techniques in a more realistic scenario, with longer time intervals and a more dynamic, less linear workloads, like it is the case in modern cloud applications. Thats the reason we chose to compare the above techniques to each other under the same conditions in this paper, analyzing also some tradeoffs regarding accuracy, like training efforts and execution time.

VII. Conclusion and Future Directions

In this paper, we investigate how predictive models for workload forecasting can be used as an alternative to resource overprovisioning to reduce provisioning costs and energy consumption still maintaining QoS levels and avoiding SLA violations. Three workload prediction techniques are analyzed, and their advantages and disadvantages are presented when applied in the context of cloud computing. Our preliminary results compare ARIMA, MLP, GRU under different cloud configurations to help administrators choose the more appropriate and efficient predictive model for their specific problem.

Our preliminary experiments show that for short time intervals all the compared techniques achieved good predictions. A disadvantage about ARIMA is that the model needs to be updated and recalculate for each new time series before the next forecast. This can be time consuming and may compromise its application in certain environments. Both neural network techniques, MLP and GRU, need fewer samples to give precise forecasts in all of the tested configurations. Based on the RMSE metric, GRU achieved a slightly better accuracy and is our recommendation for cloud scenarios with similar workload characteristics as the ones used in this work.

As future work we plan to investigate how hyperparameters can be further tuned to improve MLP and GRU results and how this tuning affects the tradeoff of each predictive model and consequently the conclusions of our comparative study.

References

[1] Amiri, M.; Mohammad-Khanli, L. "Survey on prediction models of applications for resources provisioning in cloud", *Journal of Network and Computer Applications*, 82, 2017, 93–113.

[2] Bengio, Y.; Simard, P.; Frasconi, P. "Learning long-term dependencies with gradient descent is difficult", *IEEE transactions on neural networks*, 5–2, 1994, 157–166.

[3] Bennani, M. N.; Menasce, D. A. "Resource allocation for autonomic data centers using analytic performance models". : Second International Conference on Autonomic Computing (ICAC'05), 2005, 229–240.

[4] Calheiros, R. N.; Masoumi, E.; Ranjan, R.; Buyya, R. "Workload prediction using arima model and its impact on cloud applications qos", *IEEE Transactions on Cloud Computing*, 3–4, 2015, 449–458.

[5] Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. "Learning phrase representations using rnn encoder-decoder for statistical machine translation", *arXiv preprint arXiv:1406.1078*, 2014.

[6] Diaz, G. I.; Fokoue-Nkoutche, A.; Nannicini, G.; Samulowitz, H. "An effective algorithm for hyperparameter optimization of neural networks", *IBM Journal of Research and Development*, 61–4, 2017, 9–1.

[7] Garg, S. K.; Toosi, A. N.; Gopalaiyengar, S. K.; Buyya, R. "Sla-based virtual machine management for heterogeneous workloads in a cloud datacenter", *Journal of Network and Computer Applications*, 45, 2014, 108–120.

[8] Gollapudi, S. "Practical Machine Learning". Packt Publishing Ltd, 2016.

[9] Honnappa, H.; Jain, R. "Strategic arrivals into queueing networks". : Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on, 2010, 820–827.

[10] Huang, D.; He, B.; Miao, C. "A survey of resource management in multi-tier web applications", *IEEE Communications Surveys & Tutorials*, 16–3, 2014, 1574–1590.

[11] Ita. "Traces available in the internet traffic archive." http://ita.ee.lbl.gov/html/contrib/.

[12] Jiang, Y.; Perng, C.-S.; Li, T.; Chang, R. N. "Cloud analytics for capacity planning and instant vm provisioning", *IEEE Transactions on Network and Service Management*, 10–3, 2013, 312–325.

[13] Kumar, J.; Singh, A. K. "Workload prediction in cloud using artificial neural network and adaptive differential evolution", *Future Generation Computer Systems*, 81, 2018, 41–52.

[14] Leitner, P.; Wetzstein, B.; Rosenberg, F.; Michlmayr, A.; Dustdar, S.; Leymann, F. "Runtime prediction of service level agreement violations for composite services". : Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops, 2010, 176–186.

[15] Liang, Q.; Zhang, J.; Zhang, Y.-h.; Liang, J.-m. "The placement method of resources and applications based on request prediction in cloud data center", *Information Sciences*, 279, 2014, 735–745.

[16] Manvi, S. S.; Shyam, G. K. "Resource management for infrastructure as a service (iaas) in cloud computing: A survey", *Journal of Network and Computer Applications*, 41, 2014, 424–440.

[17] Messias, V. R.; Estrella, J. C.; Ehlers, R.; Santana, M. J.; Santana, R. C.; Reiff-Marganiec, S. "Combining time series prediction models using genetic algorithm to autoscaling web applications hosted in the cloud infrastructure", *Neural Computing and Applications*, 27–8, 2016, 2383–2406.

[18] Müller, Andreas C., G. S. "Introduction to Machine Learning with Python". O'Reilly Media Inc., 2016.

[19] Nanda, S.; Hacker, T. J.; Lu, Y.-H. "Predictive model for dynamically provisioning resources in multi-tier web applications". : Cloud Computing Technology and Science (CloudCom), 2016 IEEE International Conference on, 2016, 326–335.

[20] Prevost, J. J.; Nagothu, K.; Kelley, B.; Jamshidi, M. "Prediction of cloud data center networks loads using stochastic and neural models". : System of Systems Engineering (SoSE), 2011 6th International Conference on, 2011, 276–281.

[21] Singh, S.; Chana, I. "A survey on resource scheduling in cloud computing: Issues and challenges", *Journal of grid computing*, 14–2, 2016, 217–264.

[22] Stuart, R.; Peter, N. "Artificial intelligence-a modern approach 3rd ed", 2016.

[23] Urgaonkar, B.; Pacifici, G.; Shenoy, P.; Spreitzer, M.; Tantawi, A. "Analytic modeling of multitier internet applications", *ACM Transactions on the Web (TWEB)*, 1–1, 2007, 2.

[24] Yang, J.; Liu, C.; Shang, Y.; Cheng, B.; Mao, Z.; Liu, C.; Niu, L.; Chen, J. "A cost-aware auto-scaling approach using the workload prediction in service clouds", *Information Systems Frontiers*, 16–1, 2014, 7–18.

[25] Yang, Q.; Peng, C.; Zhao, H.; Yu, Y.; Zhou, Y.; Wang, Z.; Du, S. "A new method based on psr and ea-gmdh for host load prediction in cloud computing system", *The Journal of Supercomputing*, 68–3, 2014, 1402–1417.

[26] Zhu, X.; Uysal, M.; Wang, Z.; Singhal, S.; Merchant, A.; Padala, P.; Shin, K. "What does control theory bring to systems research?", *ACM SIGOPS Operating Systems Review*, 43–1, 2009, 62–69.