Proceedings of the International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT 2023)
IEEE Xplore Part Number: CFP23CV1-ART; ISBN: 978-1-6654-7451-1

2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT) | 978-1-6654-7451-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/IDCIoT56793.2023.10053532

Cloud Resources Forecasting based on Server Workload using ML Techniques

Tejaswini Sambrajyam Janjanam
Computer Science and Engineering
V. R. Siddhartha Engineering College
Vijayawada, India
tejaswinijanjanam@gmail.com

Kavya Sharmila Siram
Computer Science and Engineering
V. R. Siddhartha Engineering College
Vijayawada, India
sharmilasiram@gmail.com

Dr. Praveen Kumar Kollu
Computer Science and Engineering
V. R. Siddhartha Engineering College
Vijayawada, India
praveen@vrsiddhartha.ac.in

***Abstract*—Provisioning of resources, automatically, according to the service demand allows cloud service providers (CSPs) to implement elastic services. It is highly essential to guarantee QoS (Quality of Service) and to fulfill SLA (Service Level Agreement), in particular for the services having strict QoS needs like web servers getting a heavy load. Over-provisioning occurs when there is low demand, under-provisioning when there is a high demand and both of these situations result in poor quality of Experience (QoE) for the users. So, an approach is required to forecast future workload accurately prior to the resource allocation so that the right amount of resources can be provisioned. Such a system can help CSPs to achieve efficient resource allocation that maximizes their economic growth and also gains users' satisfaction. This article presents an efficient workload forecasting mechanism built with the Support Vector Regression (SVR) technique to evaluate the number of resources required using web server workload time series data and queuing theory. The main goal of this technique is to forecast the future workload of the web server and predict the resources required to minimize the latency while reducing infrastructure costs and energy consumption. Normalization and outlier removal are performed for better forecasting. Various other techniques are also compared in estimating workload. Forecasting of load and prediction of resources is done with good accuracy by our proposed model better than many other models.**

***Keywords*—Server workload, Time series, Cloud resources, Support Vector Regression, Queuing model**

I. Introduction

Cloud computing converts IT infrastructure significantly computing power and data storage into a utility. Users can access these facilities from anywhere and anytime through the internet without managing these resources. And these resources are pay-as-you-go, i.e., you will only pay for what you utilize. These features are attracting many users throughout the world as it is cost-effective and zero up-front investment. A bond called service level agreement (SLA) exists between the cloud service providers (CSP) and users so that an appropriate amount of service is always present. This level of service is called quality of service (QoS). If the cloud service provider fails to provide the QoS level as agreed in SLA, there will be a penalty on the CSP. So to avoid these penalties, CSPs allocate more resources than necessary, called over-provisioning. But due to this, resources will be wasted and causing unnecessary power consumption. The service elasticity mechanism can overcome these disadvantages, which can be implemented by auto-scaling techniques. These auto-scaling techniques make automatic decisions. To aid in auto-scaling, in this paper, we determine how many resources to allocate using the previous data.

A. Basic Concepts

a) Time Series Forecasting as Supervised Learning: Supervised Machine learning is the sub-field of machine learning. In this method, labeled datasets are required for training the model. Output variable (say y) is defined as the function of input variable (say x) and this function is the form of the model.

$$y = f(x) \tag{1}$$

Time series is a sequence of data values of a particular system usually taken with equal intervals of time. In our workload data of a web server, $x_1, x_2, x_3, \ldots, x_n$ are n values of server load (requests/sec) measured on average, each hour, we try to forecast the load at successive intervals $x_{n+1}, x_{n+2}, \ldots$ Time series forecasting can be converted to supervised machine learning problem to allow application of both linear and non-linear ML algorithms on time series data. A time step (output variable) can be calculated as a function of the previous time steps (input variables). Thus, the preceding time steps predict the value at the succeeding time step. Order of observations in data must not be disturbed.

b) Auto Regression Model: In regression model, target variable (y) can be represented as a linear combination of input features (x_i).

$$y = b_0 + b_1 * x_1 + b_2 * x_2 \tag{2}$$

In (2), b_0, b_1, b_2 values are found by training the model. We can apply this method to the time series dataset with preceding time steps as input variables. Since the model uses the data calculated at preceding time steps to find the succeeding time steps it is known as the Autoregression model, defined as:

$$y_t = \delta + \sum_{i=1}^{p} \phi_i x_{t-i} + \epsilon_t \tag{3}$$

where the $\phi_1, ..., \phi_p$ are coefficients and x_{t-i} are the last p preceding time steps and ϵ_t is the white noise.

c) Moving Average Model: In Auto Regression Model we predict the succeeding time step using the preceding time steps but there may be an error in our prediction, that is there can be a difference between the predicted time step and the actual time step. Now, these past errors can be used to optimize the model. In the moving average model rather than using the preceding time steps, we use preceding errors in a regression model to predict the succeeding time step. This error is called a residual. Moving Average Model using the last q preceding time steps can be defined as:

$$y_t = \epsilon_t + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \tag{4}$$

where $\theta_1, ..., \theta_q$ are parameters of model and $\epsilon_t, \epsilon_{t-1}, ..., \epsilon_{t-q}$ are prior q white noise terms.

d) Support Vector Regression: Support Vector Regression (SVR) is a regression-based algorithm that supports linear, non-linear regressions. It is built upon the Support Vector Machine(SVM) principle. The SVM classifier predicts discrete categorical variables, but Support Vector Regressor predicts continuous variables. In SVM, we try to get a line (in case of linear data) or a hyperplane in multidimensional space(in case of non-linear data) that divides data into two classes. In multidimensional space, a **kernel** assists us in locating a hyperplane while minimizing computational cost. The type of SVM built is determined by the kernel function used and also influences the computation time required to implement the SVM. The polynomial Kernel function of degree d is represented as

$$K(x_i, x_j) = (x_i, x_j)^d \tag{5}$$

A Gaussian radial basis kernel function is represented as

$$K(x_i, x_j) = \exp(-(x_i - x_j)^2/\sigma^2) \tag{6}$$

SVR allows us to specify the model's acceptable amount of error. It focuses on minimizing the coefficients, especially the $l^2 - norm$ of the coefficient vector, rather than the squared error. Instead, the error term is taken care of by the constraints while setting the absolute error <= threshold, known as the maximum error (ϵ). ϵ is to be adjusted to achieve target accuracy for the model. The objective function is

$$\textbf{min } 1/2||w^2|| + C\sum_{i=1}^{n} |\xi_i| \tag{7}$$

and $|y_i - w_i x_i| <= \epsilon + |\xi_i|$ is a constraint, where y_i is the target variable, x_i is the input variable, w_i is coefficient, ξ_i is a deviation from the margin, C is a hyperparameter.

e) M/M/c queuing model: The system performance model used here is based on queuing theory. M/M/c queue model is mostly used in multi-server systems. When there are multiple servers that can serve a variable number of requests, this model can efficiently tell how many servers are required exactly to serve all the requests within less time. The M/M/c queue model has parameters like c, λ, μ, ρ that are explained later. And ρ can be defined as

$$\rho = \lambda/c\mu \tag{8}$$

B. Motivation

Cloud computing instantly provides users with various services like storage, networking, servers, databases, software, intelligence, and analytics over the internet. The conventional method of instantiating each virtual machine (VM) with a predefined fixed amount of resources determined by the developer leads to resource fragmentation, resource contention, or over-provisioning of resources. It may be either wasting the resources which lead to wastage of power and many other things or a violation of the SLA. However, VMs not at all require a fixed amount of resources but need resources depending on the dynamic load. So to prevent those we are following dynamic resource allocation methods in Cloud. We predict the load of the system by analyzing the past load, using this we dynamically allocate the resources.

C. Problem Statement

To predict the number of resources (web servers) required before allocation to prevent over/under-provisioning.

D. Objectives

- To forecast the future load of the cloud server.
- To calculate the optimal number of resources required.
- To compare with other forecasting models.

E. Advantages

- This model optimizes the response time of the cloud service.
- With this model over-provisioning and under-provisioning can be avoided.

F. Applications

- Can be used in public clouds like AWS cloud, google cloud
- Can be used in a private cloud set up in an organization.

G. Scope

The present scope of our project is limited to the prediction of resources (web servers) required in the Cloud.

II. Related Work

J. Gao, H. Wang, and H. Shen [1] proposed a technique to conduct the forecast sometime before the predicted time to give sufficient time for task scheduling using the predicted load. They developed a clustering-based workload prediction strategy that groups all jobs into various groups, then trains a forecasting model for each group separately to improve prediction accuracy.

Prior to resource sequences and smoothing the workload in [2] by Jing Bi, Shuang Li, Haitao Yuan, and MengChu Zhou, the standard deviation is reduced using a logarithmic procedure. The design of a deep learning integrated technique for time series prediction is done to produce a high-quality workload forecast for which it utilizes network models (grid-long short-term memory networks(LSTMs) and bi-directional networks).

Sima Jeddi and Saeed Sharifian [3] proposed a new wavelet transform, GMDH-based model for workload prediction under dynamically changing conditions. To better categorize, and eliminate nonlinearity in each scale, they separated various time-frequency scales and employed a tiny nonlinear GMDH model for each scale. Finally, they ensembled the outcomes of each scale's prediction with the ELM algorithm to generate a predicted sample of the final time series for the workload.

A Self Directed Workload Forecasting method (SDWF) for cloud data centers (CDC) was proposed by Jitendra Kumar, Ashutosh Kumar Singh, and Rajkumar Buyya [4]. For training purposes, the model adopted a heuristic strategy that was inspired by nature. The approach can be adjusted such that it can identify where its previous forecasts went wrong and make necessary adjustments to its future predictions.

Hisham A. Kholidy [5] presented a Swarm Intelligence Based Prediction Approach (SIBPA) that integrates linear and nonlinear prediction models with the help of ARIMA and MSVR, respectively. The resource requirements of a cloud customer in terms of CPU, memory, and disk storage consumption are predicted by SIBPA more precisely. The SIBPA can also forecast response times and throughput, which helps cloud users in making good decisions on scaling.

To accurately forecast impending workload, Jitendra Kumar, Deepika Saxena, Ashutosh Kumar Singh, and Anand Mohan, [6] developed a neural network model for workload prediction. They constructed the BaDE learning algorithm, a new differential evolution technique that allows adaptation during the phases of mutation and crossover, to train the model. Compared to cutting-edge techniques, their forecasting model has attempted to find unexpected peaks of undetected workload patterns.

Yonghua Zhu, Weilin Zhang, Yihai Chen, and Honghao Gao, [7] presented a new method for workload forecasting that makes use of an attention-based LSTM encoder-decoder network. The attention mechanism is then incorporated into the decoder network to improve the model's capacity for forecasting batch workload. They also presented a scroll prediction approach, which divides a long-term forecasting job into multiple smaller jobs, to minimize the error that occurs during long-term forecasting.

For handling dynamic and extremely changeable cloud workloads, I. K. Kim, W. Wang, Y. Qi, and M. Humphrey, [8] introduced CloudInsight, an online workload prediction system. CloudInsight uses several local predictors to build an ensemble forecasting model by dynamically finding the appropriate contributions of each local predictor. They structured this case as a multi-class regression condition using an SVM classifier in order to estimate the weights.

III. Methodology

A. Modules

a) Forecasting Module: In this module firstly, we collect cloud server load data containing DateTime and hits columns. After preprocessing, we applied various forecasting models to that data based on:
1. The last value
2. Moving Average
3. Auto Regression
4. SVR with polynomial kernel
5. SVR with normalized polynomial kernel
6. SVR with RBF kernel ($\gamma = 0.2$)
7. SVR with RBF kernel ($\gamma = 0.6$)
8. SVR with RBF kernel ($\gamma = 1.0$)

Time-series forecasting model we considered here is the SVR model. To analyze the time series behavior, the training dataset must be large enough. So, here, we took data for 4 weeks (672 hours) hour-wise data. To determine a suitable lag period, we carefully observed the pattern of input data repeating every 24 hours. So, 24 is taken as lag. 1 hour is set as our forecasting horizon due to which forecast for the next one hour is done based on the previous few observations. The reason for selecting this horizon is that a higher forecasting horizon leads to lesser prediction accuracy, that's why it's better to make hourly new predictions. To achieve good estimation accuracy, γ(hyperparameter), C(regularized risk), ϵ(Vapnik loss function) are also to be chosen carefully. Then, for a test interval of 24 hours, we obtained hourly predictions.

b) Resource Prediction Module: In this module, we have used M/M/c queuing model for the evaluation of the performance (calculation of resources required) of our forecast models. The chosen parameters of the M/M/c model are as follows:

- For each unit of time, how many requests can be processed by each backend server is given by Service rate. It is denoted by μ. We took it as 250 requests/s as it is the general throughput of a normal server.
- The server's mean load (requests/s) for every hour is represented by the arrival rate, denoted by λ. We took into account the load predicted by each forecasting model for a test interval of 24 hours and the server's actual load.
- To ensure system stability, the system utilization factor $\rho < 1$ is better.

The required number of servers (Predicted allocation) is then calculated based on the forecasted load. We can also compare that with the optimal number of resources based on actual load (Actual allocation) because the server's real hourly load of the test data set is known. It can be figured out whether under-provisioning (Predicted allocation $<$ Actual allocation) has occurred or over-provisioning (Predicted allocation $>$ Actual allocation) has occurred or we provisioned correctly (Predicted allocation = Actual allocation), for all forecasting models. Fig. 13 is constructed based on this.

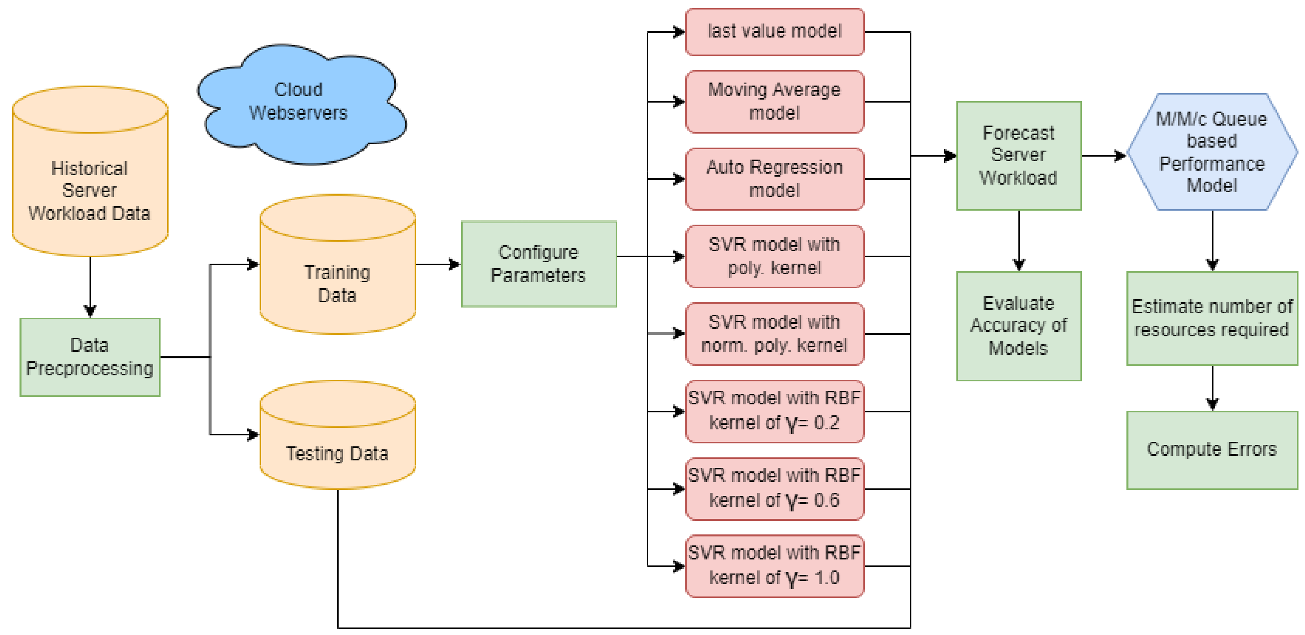

Historical Server Workload Data
Cloud Webservers
Data Precprocessing
Training Data
Testing Data
Configure Parameters
last value model
Moving Average model
Auto Regression model
SVR model with poly. kernel
SVR model with norm. poly. kernel
SVR model with RBF kernel of γ= 0.2
SVR model with RBF kernel of γ= 0.6
SVR model with RBF kernel of γ= 1.0
Forecast Server Workload
Evaluate Accuracy of Models
M/M/c Queue based Performance Model
Estimate number of resources required
Compute Errors

Fig. 1. Proposed system

B. Algorithms

a) Support Vector Regression Algorithm:
1. Import the necessary libraries.
2. Load data from the CSV file into a data frame using Pandas.
3. Visualize the available input data with some plots.
Prepare the data for training:
4. Create the training data with required 'hits' and date columns with required time periods.
5. Scale training and testing data using MinMaxScaler (here it scales hits column to [0,1] range to lower the distance between them).
6. Alter training and testing data to a 2D tensor consisting of timesteps.
Implement SVR:
7. Create an SVR model by passing the required parameters: kernel, γ, c, and ϵ
8. Train the model using the fit method.
9. Predict values of test data using predict method.
10. Plot time series forecast vs original values to observe the difference.
11. Find out MAPE to evaluate the model.

b) Auto Regression Algorithm:
1. Import the required libraries.
2. Define the training and test data set.
3. Determine the lag periods (or order p) by observing ACF and PACF plots.
4. Train the Autoregression Model.
5. Make predictions.
6. Plot the Result.

c) Moving Average Algorithm:
1. Gather the data.
2. Test for stationarity.
3. Find q. The ACF plot will show significant autocorrelation coefficients up until lag q, after which all coefficients will be non-significant.
4. Apply MA to make time series forecast.
5. Plot results

TABLE I: Parameters of different forecasting models

Forecasting Model	**Parameters**
Last Value	–
Moving Average	q = 4
Auto Regression	p = 24
SVR with polynomial kernel	degree = 1, C = 0.78, ϵ = 0.05
SVR with norm. polynomial kernel	degree = 2, C = 0.78, ϵ = 0.05
SVM with RBF kernel(low)	γ = 0.2, C = 0.78, ϵ = 0.05
SVM with RBF kernel(medium)	γ = 0.6, C = 0.78, ϵ = 0.05
SVM with RBF kernel(high)	γ = 1.0, C = 0.78, ϵ = 0.05

C. Performance Comparision

We used three error measures: Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), in this work to compare the models.

$$MAPE = 1/n \sum_{i=1}^{n} |a_i - p_i/a_i| \tag{9}$$

$$MAE = 1/n \sum_{i=1}^{n} |a_i - p_i| \tag{10}$$

$$RMSE = \sqrt{1/n \sum_{i=1}^{n} (a_i - p_i)^2} \tag{11}$$

Here, a_i is the actual value, and p_i is the predicted value.

IV. Results And Analysis

This section details the results and outputs of the system. Fig. 2 shows server workload data taken as input.

```
id           int64
timestamp    object
hits         int64
dtype: object
```

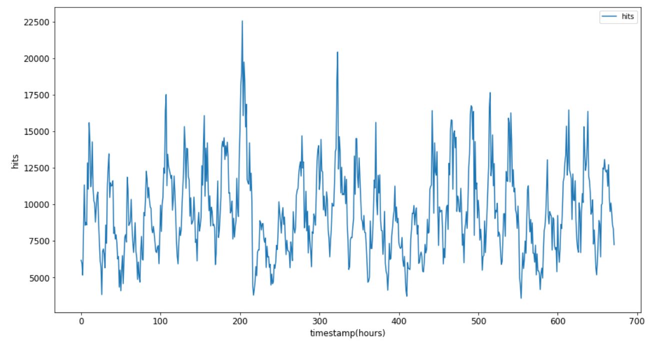
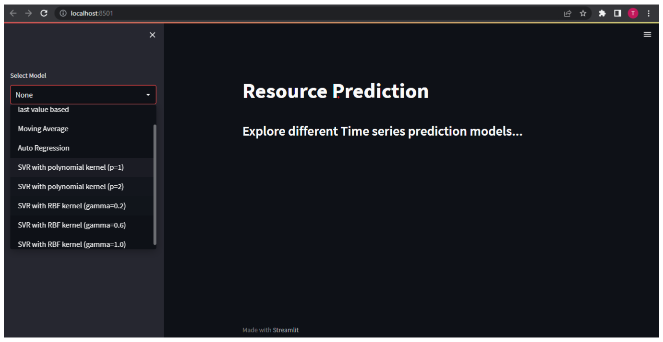


Fig. 2. Time series training data

Fig. 3 shows a webpage designed using Streamlit in Python to compare the 8 forecasting models and compute resources required and compare the errors. This is also deployed in the Streamlit cloud and can be accessed using: https://teja99-tech-mini-project-main-jy4mke.streamlit.app/


localhost:8501
Select Model
None
last value based
Moving Average
Auto Regression
SVR with polynomial kernel (p=1)
SVR with polynomial kernel (p=2)
SVR with RBF kernel (gamma=0.2)
SVR with RBF kernel (gamma=0.6)
SVR with RBF kernel (gamma=1.0)
Made with Streamlit

Resource Prediction

Explore different Time series prediction models...

Fig. 3. Webpage to show resource prediction results

Fig. 4-11 show 24hr forecasts of different models used.

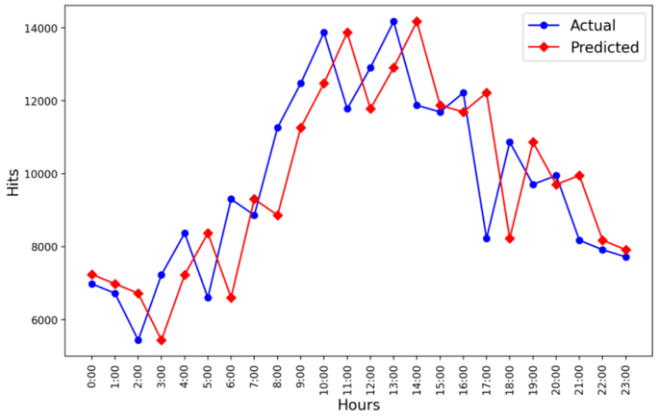


Fig. 4. Load prediction by last value model

We got MAPE as 14.50% which is very high as this model just outputs the previous value.

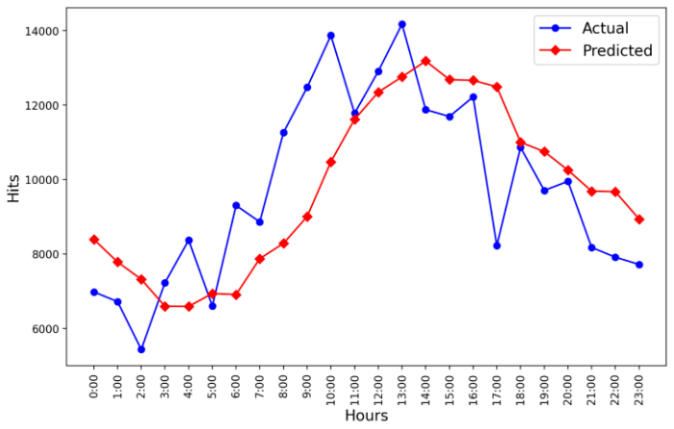


Fig. 5. Load prediction by Moving Average model

Moving Average model got the highest MAPE value as 20.78% and we can also observe heavy deviation of predicted values from actual ones.

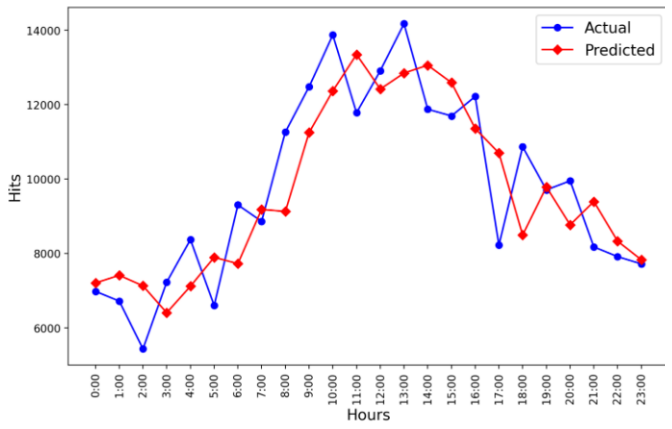


Fig. 6. Load prediction by Auto Regression model

Auto Regression model's MAPE value is 12.00%, a lot better when compared with Moving Average.

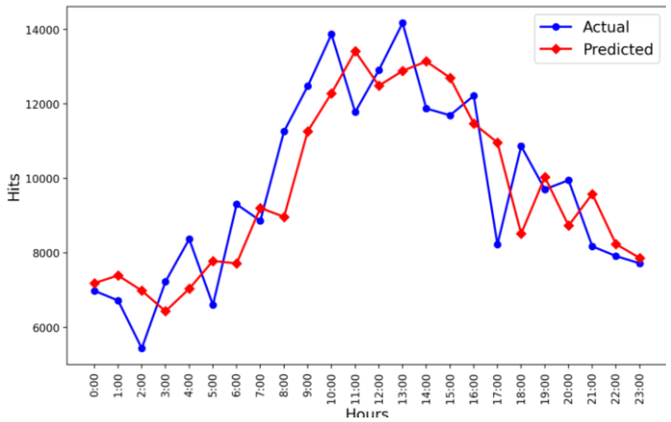


Fig. 7. Load prediction by SVR model with poly. kernel

Its MAPE value is 12.26%.

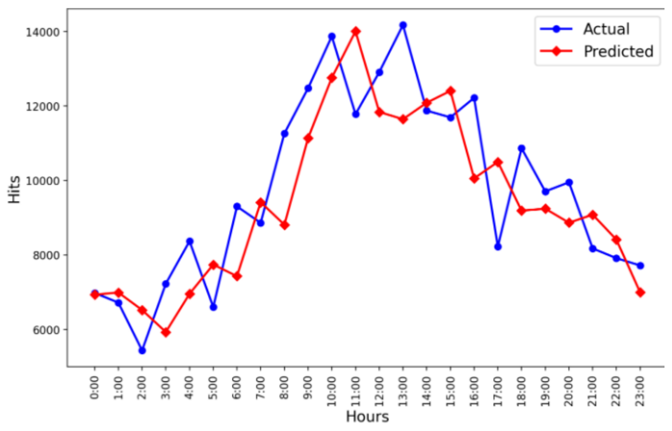


Fig. 8. Load prediction by SVR model with norm. poly. kernel

Its MAPE value is 13.20%.

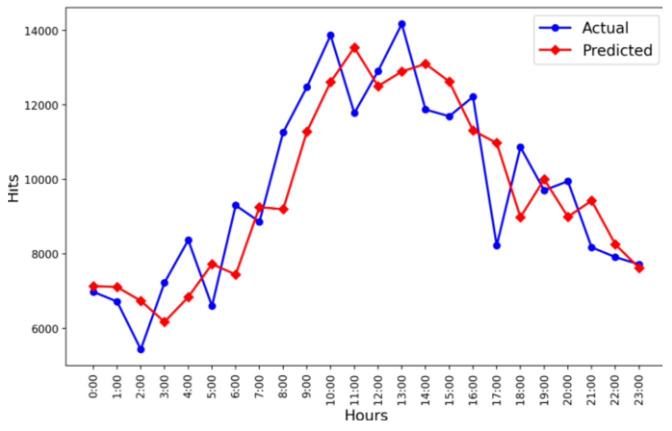


Fig. 9. Load prediction by SVR model with RBF kernel of γ = 0.2

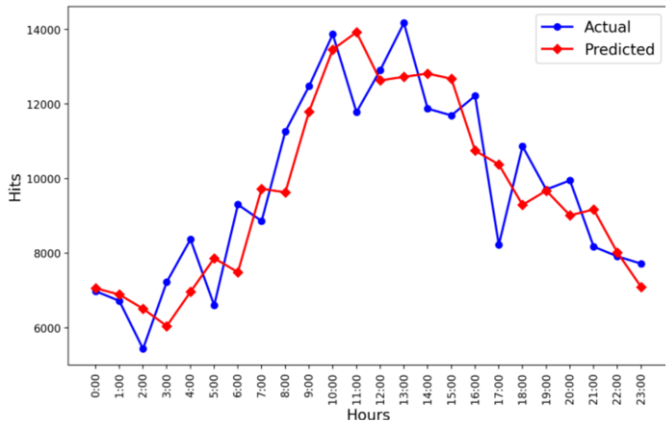


Fig. 10. Load prediction by SVR model with RBF kernel of γ = 0.6

RBF kernel models performed very well in forecasting. When γ = 0.2, MAPE = 11.77%, when γ = 0.6, MAPE = 10.91% (least of all models), when γ = 1.0, MAPE = 11.32%.

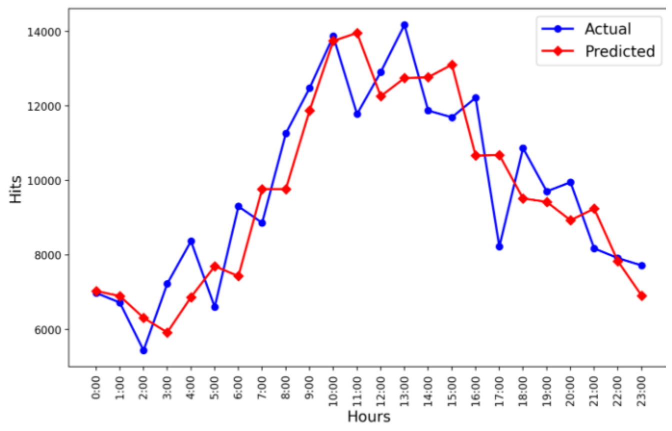


Fig. 11. Load prediction by SVR model with RBF kernel of γ = 1.0

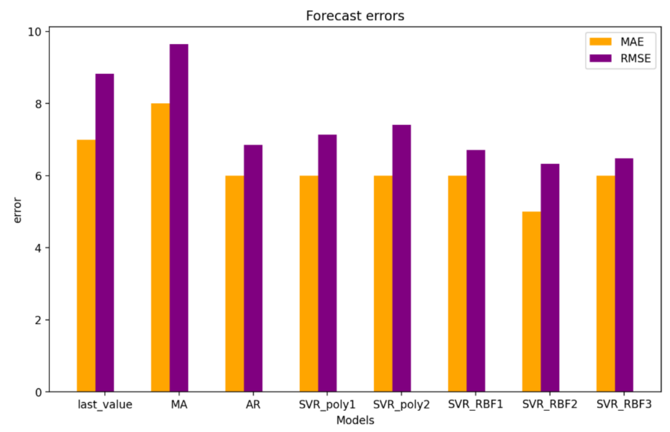
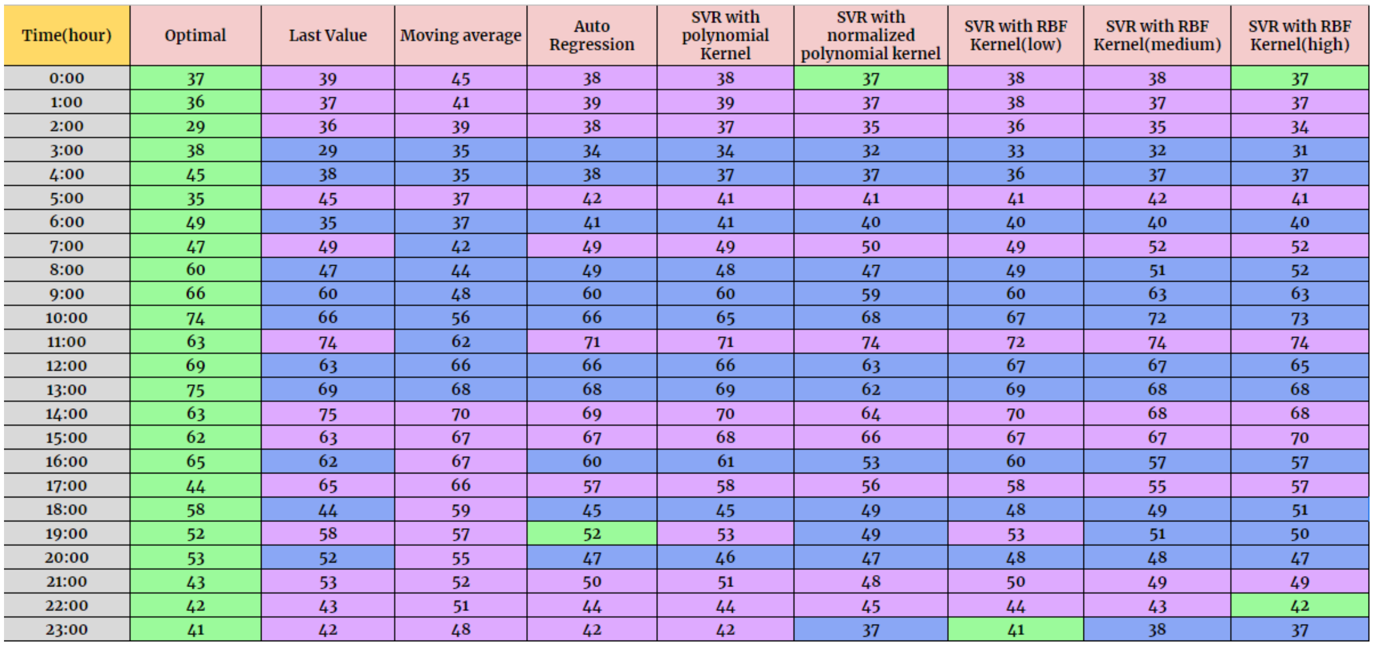


Fig. 12. Load forecast errors of all models

The comparison of values forecasted with 8 forecast models and calculated using queueing model is shown in Fig. 13. Green indicates correctly provisioned, purple indicates over-provisioned, and blue indicates under-provisioned. The Actual allocation results calculated with the actual server load are displayed in the "Optimal" column. Predicted allocation results calculated with forecasted server loads are displayed in columns 1 through 8. It can be seen that the predictions made using SVR forecasting models (last 5 columns), in most situations, are very near to the ideal value. The basic forecasting models (Last value, Moving Average, Auto Regression) have variable values. This shows that SVR-based models are good at forecasting than others(used in this work). MAE (10) and RMSE (11) errors are depicted in Fig. 12. The SVR forecasting model with RBF Kernel, γ = 0.6 is the ideal scenario in this.

V. Conclusion

This research work has proposed a resource prediction system based on forecasting with ML techniques and performance queues, that helps in the provisioning of resources by maintaining elasticity in cloud services, with the goal of minimizing response time. Based on historical data, this system predicts the load of a web server using SVR. We configured the SVR model's parameters before applying them. By calculating the



Time(hour)	Optimal	Last Value	Moving average	Auto Regression	SVR with polynomial Kernel	SVR with normalized polynomial kernel	SVR with RBF Kernel(low)	SVR with RBF Kernel(medium)	SVR with RBF Kernel(high)
0:00	37	39	45	38	38	37	38	38	37
1:00	36	37	41	39	39	37	38	37	37
2:00	29	36	39	38	37	35	36	35	34
3:00	38	29	35	34	34	32	33	32	31
4:00	45	38	35	38	37	37	36	37	37
5:00	35	45	37	42	41	41	41	42	41
6:00	49	35	37	41	41	40	40	40	40
7:00	47	49	42	49	49	50	49	52	52
8:00	60	47	44	49	48	47	49	51	52
9:00	66	60	48	60	60	59	60	63	63
10:00	74	66	56	66	65	68	67	72	73
11:00	63	74	62	71	71	74	72	74	74
12:00	69	63	66	66	66	63	67	67	65
13:00	75	69	68	68	69	62	69	68	68
14:00	63	75	70	69	70	64	70	68	68
15:00	62	63	67	67	68	66	67	67	70
16:00	65	62	67	60	61	53	60	57	57
17:00	44	65	66	57	58	56	58	55	57
18:00	58	44	59	45	45	49	48	49	51
19:00	52	58	57	52	53	49	53	51	50
20:00	53	52	55	47	46	47	48	48	47
21:00	43	53	52	50	51	48	50	49	49
22:00	42	43	51	44	44	45	44	43	42
23:00	41	42	48	42	42	37	41	38	37

Fig. 13. Comparison of resources predicted by all models

RMSE and MAE errors of test data, it is found that SVR-based models are better at forecasting than basic models. SVR with RBF kernel, in particular, provides the best allocation results in terms of over-provisioned resources. Furthermore, the proposed mechanism combines the forecasting model with the performance model based on M/M/c queues. This system helps cloud service providers to predict sufficient resources required, minimizes latency, meets the user's Service Level Agreement. We will try to develop an allocation model for these predicted resources as a part of future work.

REFERENCES

[1] J. Gao, H. Wang, and H. Shen, "Machine Learning based workload prediction in cloud computing," 29th International Conference on Computer Communications and Networks (ICCCN), 2020, https://doi.org/10.1109/ICCCN49398.2020.9209730

[2] Jing Bi, Shuang Li, Haitao Yuan, MengChu Zhou, "Integrated deep learning method for workload and resource prediction in cloud systems", Neurocomputing, ISSN 0925-2312, 2021, https://doi.org/10.1016/j.neucom.2020.11.011

[3] Sima Jeddi, Saeed Sharifian, "A hybrid wavelet decomposer and GMDH-ELM ensemble model for Network function virtualization workload forecasting in cloud computing", Applied Soft Computing, ISSN 1568-4946, 2020, https://doi.org/10.1016/j.asoc.2019.105940

[4] Jitendra Kumar, Ashutosh Kumar Singh, Rajkumar Buyya, "Self-directed learning based workload forecasting model for cloud resource management", Information Sciences, ISSN 0020-0255, 2021, https://doi.org/10.1016/j.ins.2020.07.012

[5] Hisham A. Kholidy, An Intelligent Swarm Based "Prediction approach for predicting cloud computing user resource needs", Computer Communications, ISSN 0140-3664, 2020, https://doi.org/10.1016/j.comcom.2019.12.028

[6] Jitendra Kumar, Deepika Saxena, Ashutosh Kumar Singh, Anand Mohan, "BiPhase adaptive learning-based neural network model for cloud datacenter workload forecasting", Soft Computing, 2020, https://doi.org/10.1007/s00500-020-04808-9

[7] Yonghua Zhu, Weilin Zhang, Yihai Chen, Honghao Gao, "A novel approach to workload prediction using attention-based LSTM encoder-decoder network in cloud environment", EURASIP Journal on Wireless Communications and Networking, 2019, https://doi.org/10.1186/s13638-019-1605-z

[8] I. K. Kim, W. Wang, Y. Qi, and M. Humphrey, "Forecasting cloud application workloads with CloudInsight for predictive resource management", IEEE Transactions on Cloud Computing, 2022, https://doi.org/10.1109/TCC.2020.2998017

[9] Cédric St-Onge, Souhila Benmakrelouf, Nadjia Kara, Hanine Tout, Claes Edstrom, Rafi Rabipour, "Generic SDE and GA-based workload modeling for cloud systems", Journal of Cloud Computing: Advances, Systems and Applications, 2021, https://doi.org/10.1186/s13677-020-00223-5

[10] Ehsan Golshani, Mehrdad Ashtiani, "Proactive auto-scaling for cloud environments using temporal convolutional neural networks", Journal of Parallel and Distributed Computing, ISSN 0743-7315, 2021, https://doi.org/10.1016/j.jpdc.2021.04.006

[11] Gadhavi, L.J., Bhavsar, M.D., "Efficient resource provisioning through workload prediction in the cloud system", In Zhang, YD., Mandal, J., So-In, C., Thakur, N. (eds) Smart Trends in Computing and Communications, Smart Innovation, Systems and Technologies, Springer, 2020, https://doi.org/10.1007/978-981-15-0077-0_33

[12] Boyun Liu, Jingjing Guo, Chunlin Li, Youlong Luo, "Workload forecasting based elastic resource management in edge cloud", Computers & Industrial Engineering, ISSN 0360-8352, 2020, https://doi.org/10.1016/j.cie.2019.106136

[13] A. Vashistha and P. Verma, "A literature review and taxonomy on workload prediction in cloud data center," 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2020, https://doi.org/10.1109/Confluence47617.2020.9057938

[14] Masdari M., Khoshnevis A., "A survey and classification of the workload forecasting methods in cloud computing", Cluster Computing, 2020, https://doi.org/10.1007/s10586-019-03010-3

[15] S. Wang, Y. Yao, Y. Xiao, H. Chen, "Dynamic resource prediction in cloud computing for complex system simulation: A probabilistic approach using stacking ensemble learning," 2020 International Conference on Intelligent Computing and Human-Computer Interaction (ICHCI), 2020, https://doi.org/10.1109/ICHCI51889.2020.00050