Predictive Analysis for Cloud Resource Management

Thesis submitted in partial fulfillment of requirements
for the degree of
Master of Computer Technology
of
Department of Computer Science and Engineering
Jadavpur University

by

Subhajit Saha

Registration No.:149843 of 2019-2020
Examination Roll No.: M6TCT22010

Under the supervision of

Prof. Sarbani Roy

Department of Computer Science and Engineering
Jadavpur University
Kolkata, West Bengal, India
2022

Certificate from the Supervisor

This is to certify that the thesis entitled "Predictive Analysis for Cloud Resource Management" has been satisfactory completed by Subhajit Saha (Registration No.: 149843 of 2019-2020, Examination Roll Mo.: M6TCT22010, Class Roll No.: 001910504008). It is a bona-fide piece of work carried out under my supervision and guidance at Jadavpur University, Kolkata for the partial fulfillment of the requirements for the awarding of the Master of Computer Technology degree of the Department of Computer Science and Engineering, Faculty of Engineering and Technology, Jadavpur University during the academic year 2019 - 2022.

.....
Prof. Sarbani Roy
Department of Computer Science and Engineering
Jadavpur University (Supervisor)

Forwarded By:

.....
Prof. Anupam Sinha
Head
Department of Computer Science and Engineering
Jadavpur University

.....
Prof. Chandan Mazumdar
DEAN
Faculty of Engineering and Technology
Jadavpur University

Department of Computer Science and Engineering
Faculty of Engineering and Technology
Jadavpur University, Kolkata – 700032

Certificate of Approval

This is to certify that the thesis entitled "Predictive Analysis for Cloud Resource Management" is a bona-fide record of work carried out by Subhajit Saha (Registration No.: 149843 of 2019-2020, Examination Roll No.: M6TCT22010, Class Roll No.: 001910504008) in a partial fulfillment of the requirements for the award of degree of Master of Computer Technology in the Department of Computer Science and Engineering, Jadavpur University. It is understood that by this approval, the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein but approve the thesis only for the purpose of which it has been submitted.

Examiners:

…….. ……
(Signature of Examiner) (Signature of Supervisor)

Department of Computer Science and Engineering
Faculty of Engineering and Technology
Jadavpur University, Kolkata – 700032

Declaration of Originality and Compliance of Academic Ethics

I hereby declare that the thesis entitled "Predictive Analysis for Cloud Resource Management" contains literatures survey and original research work by the undersigned work, as a part of his degree of Master of Computer Technology in the Department of Computer Science and Engineering, Jadavpur University. All information have been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rule and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Name: Subhajit Saha
Examination Roll No.: M6TCT22010
Registration No.- 149843 of 2019-2020
Thesis Title: Predictive Analysis for Cloud Resource Management

……
Signature of the Candidate

ACKNOWLEDGEMENT

I am passed to express my gratitude & regards towards my thesis Guide Prof. Sarbani Roy, Department of Computer Science and Engineering, Jadavpur University, without whose valuable guidance, inspiration and attention towards me, pursuing my thesis would have been impossible.

I am grateful towards all the members of DST FIST LAB, Computer Science and Engineering Department, Jadavpur University, for cooperating with me in all possible ways and providing me with a good working environment throughout the duration of the work.

I am also thankful to Prof. Anupam Sinha, Head, Department of Computer Science and Engineering, Jadavpur University, and Prof. Chandan Mazumdar, DEAN, Faculty of Engineering and Technology, Jadavpur University.

I thank to Mr. Sounak Banerjee, Research Scholar, Computer Science and Engineering Department, Jadavpur University for cooperating with me in all possible ways and providing me with a good working environment throughout the duration of the work.

Last but not the least, I express my regards towards my friends & family for bearing with me and for being a source of constant motivation during the entire term of the work.

Subhajit Saha
MTCT Final Year
Exam Roll No. - M6TCT22010
Registration No.- 149843 of 2019-2020
Department of Computer Science and Engineering
Jadavpur University

Contents

Abstract

Over the past few years, there has been a considerable increase in the strain placed on the cloud server due to the rapid growth in demand for cloud resources. Resource provisioning is one of the complex issues that arise in a cloud system. Resources should be distributed dynamically based on the application's changing demand. Over-provisioning results in increased costs and energy waste. Contrarily, under-provisioning leads to SLA violations and a drop in service quality (QoS). As a result, resource allocation should be as close to the applications' current demand as possible. So, the Workload prediction is essential in determining the precise resources needed for an application to run successfully on the cloud. If a cloud computing model uses its resources in the most effective way possible, it is considered efficient. By using the Statistical models along with the Machine Learning model, this work shows a comparative analysis for predicting the workload of the virtual machines. Result shows that the machine learning model give better results than the statistical models.

1. Chapter-I

1.1 Introduction:

Cloud computing has revolutionized the IT industry in recent years. It is a paradigm where processing, storage, and network capacity are made available to users in an on demand manner through virtualization on a shared physical infrastructure. With the increasing of rapid demand for cloud resources, load on the cloud server is increased drastically in the past few years. Lots of companies shift their services and applications to cloud platforms. It can reduce the overall operational cost for large and small companies due to sharing of computing resources. It can also substantially lower the energy consumption when compared to conventional private computing facilities, which helps achieving a greener world.

Cloud computing provides a number of large computing infrastructures for large scale data centers(DC), which contains a number of physical machines(PM) with multiple virtual machines(VM) running on them. So the resource management is one the challenges for successful cloud environment. The key technology behind cloud computing is virtualization which a vital role in managing and coordinating access to the resource pool via a software layer called Virtual Machine Monitor (VMM) or hypervisor that hides the details of the physical resources and provides virtualized resources for high level applications. Also, it virtualizes all of the resources of a given PM allowing several VMs to share its resources. Virtualization also allows gathering several virtual machines into a single physical server using a technique called VM consolidation which provides significant benefits to cloud computing by facilitating better use of the available data center resources. . It can be performed either statically or dynamically. In static VM consolidation, the VMM allocates the physical resources to the VMs based on peak load demand. This leads to resource wastage because the workloads are not always at peak. On the other hand, in case of dynamic VM consolidation, the VMM changes the VM capacities according to the current workload demands. This helps in utilizing the data centers resources efficiently.

In dynamic VM consolidation, the VMs can be dynamically reallocated using live migration according to the current PM resource demand to minimize the number of

active physical servers, referred to as hosts, required to handle the workload. The idle hosts are switched to low power modes with fast transition times to eliminate the static power (power consumed while there is no circuit activity so it is considered as wasted energy) and reduce the overall energy consumption. The hosts are reactivated with the increase of the resource demand to avoid violating QoS requirements in terms of service-level agreements (SLA). This approach has basically two objectives, namely minimization of energy consumption and maximization of the quality of service (QoS) delivered by the system, which forms an energy-performance tradeoff.

The challenge of resources prediction is the estimation of correct amount of needed resources, has been tackled through proactive prediction approaches which predicts the future needed resources in a way the resource provisioning manager has sufficient time to allocate resources before occurring the bottleneck situation. Resource prediction will help to achieve the SLA signed with the customers or users. Effective prediction of resource can facilitate load balancing and proactive job scheduling across DCs which results in improved resource utilization, lowering DC costs, and improved job performance.

1.2 Cloud Types:

With the use of cloud computing, IT departments and developers are able to concentrate on what matters most while avoiding undifferentiated tasks like purchasing, upkeep, and capacity planning. As cloud computing has become more and more popular, a variety of models and deployment techniques have arisen to fulfill the various needs of users.

1.2.1 Deployment Models

Based on ownership, scale, and access as well as the nature and function of the cloud, the cloud deployment model determines the particular sort of cloud environment. A cloud deployment model specifies the location of the servers that using and it manages them. In **[1]** the design of the cloud infrastructure, what it can modify, and whether it will receive services or must develop everything from scratch. The forms of cloud deployment also determine the connections between the infrastructure and the consumers.

Different types of cloud computing deployment models are:

1. Public cloud
2. Private cloud
3. Hybrid cloud
4. Community cloud

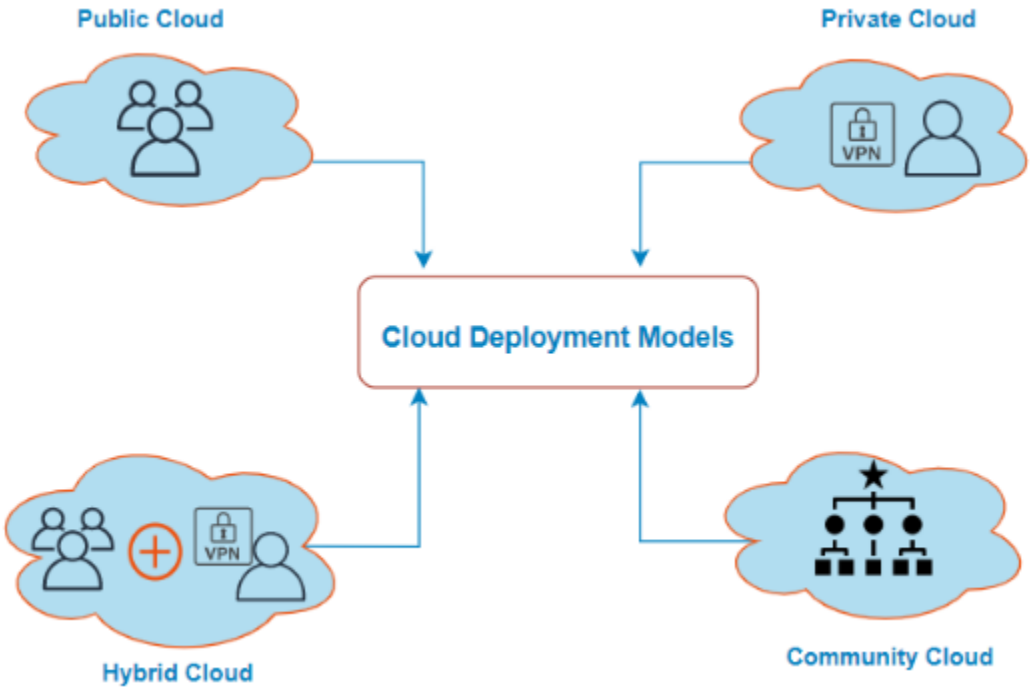


Fig. 1 Cloud Deployment Models

Public Cloud:

System and service access is made available via the public cloud for everyone. Being accessible to all users could make the public cloud less secure. The term "public cloud" refers to a cloud infrastructure service that is made available to the general public or significant business sectors over the internet. In this cloud model, the organization providing the cloud services owns the infrastructure, not the user. Customers and users can readily access systems and services thanks to this form of cloud hosting. This type of cloud computing is a great illustration of cloud hosting, in which service providers give services to a range of clients. The agreement provides free storage backup and retrieval services in exchange for a subscription, or on a per-use basis. Example: Google App Engine etc.

Private Cloud:

The public cloud deployment model contrasts sharply with the private cloud deployment model. A single user is in a one-on-one setting (customer). Users don't have to let anyone else use your hardware. How users manage all of the hardware makes the difference between a private and public cloud. The ability to access systems and services inside a certain border or organization is frequently referred to as the "internal cloud." The cloud platform is put into use in a highly secure environment that is hosted in the cloud, guarded by robust firewalls, and managed by an organization's IT department. Greater flexibility and control over cloud resources are provided by the private cloud.

Hybrid Cloud

Hybrid cloud computing provides the best of both worlds by linking the public and private realms with a layer of proprietary software. By using a hybrid solution, users may host the app in a secure location and benefit from the financial advantages offered by the public cloud. Depending on needs, organizations might use a combination of two or more cloud deployment methods to move data and applications between various clouds.

Community Cloud:

It makes it possible for a number of organizations to access systems and services. It is a distributed system made by combining the capabilities of various clouds in order to cater to the unique requirements of a community, sector, or company. If an organization has common goals or responsibilities, it may decide to share the infrastructure of the community. Usually, it is run by a third party or a group of one or more local group.

1.2.2 Service Model:

Using a network of remote servers hosted on the Internet to store, manage, and process data instead of a local server or a personal computer is known as cloud computing. These businesses are known as cloud providers and often charge for cloud computing services based on consumption. Cloud computing is built on grids and clusters. In **[2]** shows the different cloud service models given in Fig. 2.

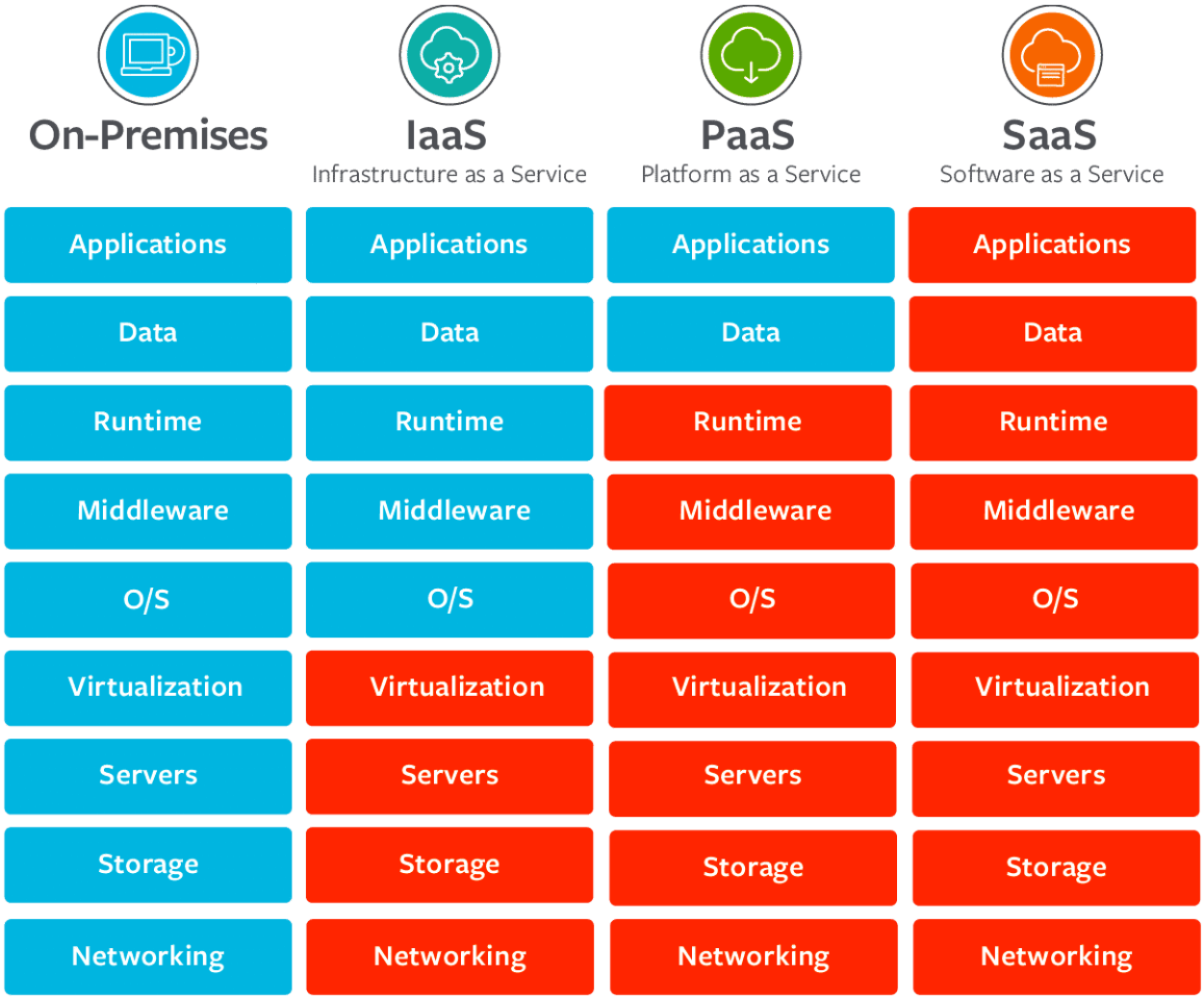


Fig. 2 Cloud Service Model

Software as a Service (SaaS)

SaaS is a method of delivering services and applications through the Internet. Users just access software via the Internet, relieving ourselves from the difficult software and hardware management, rather than installing and maintaining it. We no longer need to install and maintain apps on our own computers or in data centers, saving us money on both hardware and software upkeep.

Pay-as-you-go cloud service providers offer SaaS, which offers a comprehensive software solution. The majority of SaaS applications may be used immediately

from a web browser without the need for any downloads or installations. SaaS applications are also referred to as hosted software, web-based software, and on-demand software.

Platform as a Service (PaaS)

PaaS is a subcategory of cloud computing that offers a platform and environment to developers so they can create online apps and services. Users can easily access PaaS services via a web browser because they are hosted in the cloud. On its own infrastructure, a PaaS provider hosts the hardware and software. Because of this, PaaS relieves users from needing to install hardware and software locally in order to create or execute a new application. As a result, the hardware has no bearing on how the application is developed and deployed.

The consumer has control over the deployed application and perhaps the configuration options for the environment where the applications are hosted, but not the network, servers, operating systems, or storage that make up the underlying cloud infrastructure.

Infrastructure as a Service (IaaS)

IaaS is a service delivery model that outsources the provision of computer infrastructure to support various operations. IaaS is typically a service where infrastructure, such as networking hardware, devices, databases, and web servers, is provided as outsourcing to businesses. IaaS customers typically pay by the hour, week, or month and pay per user. Customers may also be charged based on how much virtual machine space they use by some providers.

For developing such applications and services, deploying development tools, databases, etc., it merely provides the underlying operating systems, security, networking, and servers.

1.3 Motivation:

The load on the cloud server has significantly increased over the past few years due to the rapidly growing demand for cloud resources. The provisioning of resources is one of the difficult issues in the cloud environment. According to the application's changing demand, the resources should be distributed dynamically. Energy waste and expenditures rise as a result of over provisioning. Under-provisioning, on the other side, results in SLA violations and a decline in the quality of service (QoS). So, the resource allocation should be close to the current demand of applications as much as possible. Therefore, Workload prediction plays a vital role in estimating the actual resource required for successful execution of an application on cloud. A cloud computing model is efficient if its resources are utilized in best possible way and such an efficient utilization can be achieved by employing and maintaining proper management of cloud resources.

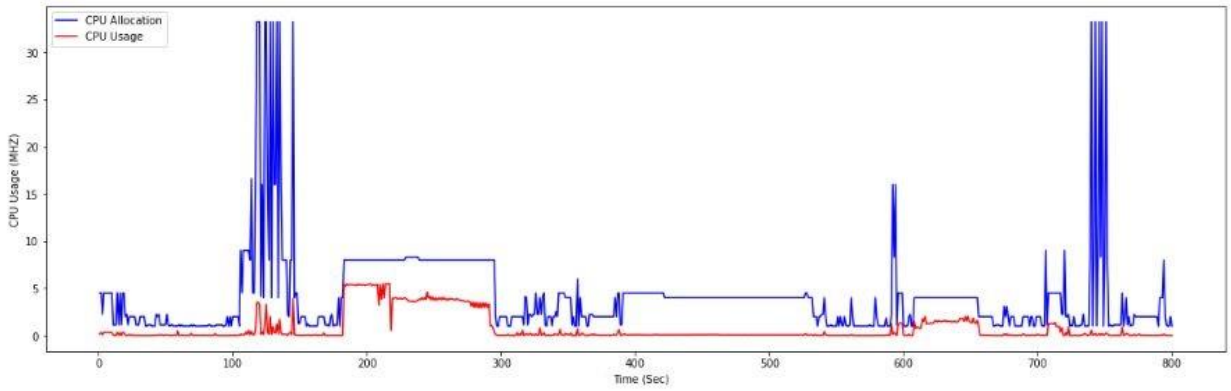


Fig 3. Average CPU capacity provisioned & CPU usage of 1250 VMs

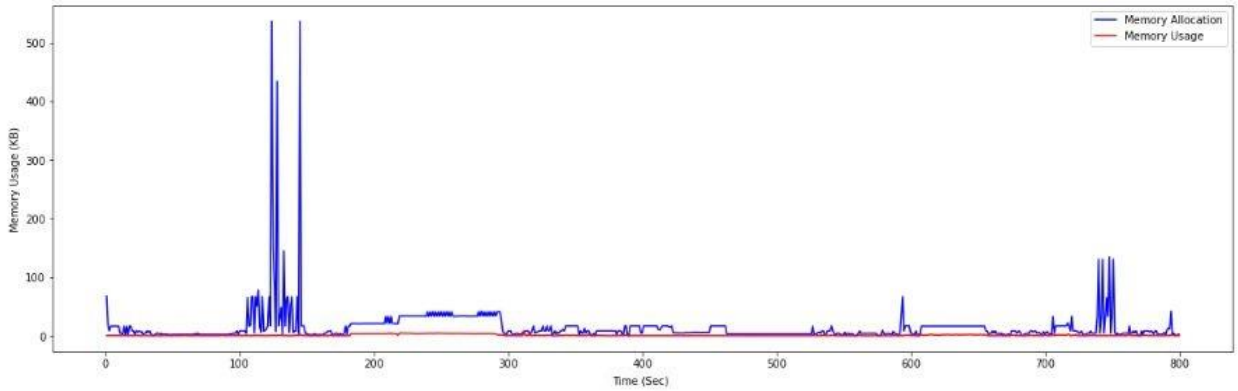


Fig 4. Average Memory capacity provisioned & Memory usage of 1250 VMs

One of the most critical and important aspect in managing the cloud infrastructure is the Workload prediction. The resources requires for every application to complete its execution come in the virtual form. The Cloud Data Centers (CDC) comprises of various resources like CPU, Memory, bandwidth, software, etc. which are allocated to the users on demand to complete their task execution. In some previous works, it shows that resources provisioned to execute an application are always greater than actual resources required to complete it **[3]**. The reason for over provisioning of resources is to avoid SLA violations and to achieve QoS satisfaction. In most of the cases, the resources are being wasted in the process of allocation.

The design of efficient resource prediction model is the major challenge in cloud environment. One of the challenge is to supply good quality of services in cloud by allocating the resources to the cloud applications via elasticity load balancing, while taking minimal infrastructure cost and response time. The crucial task is to select the appropriate resources for particular cloud applications in cloud environment. Cloud provider allocates resources to incoming requests according to the predefined resource provisioning policies. The prediction model must be investigating all dimensions of the application to get effective results in prediction.

Some of the Challenges of the prediction approaches are as follows:

• Estimation of the future behavior pattern of the application in different manners such as number of arrival requests, number of virtual machines or physical machines, SLA violations, resources utilizations etc., is a challenging task.

• Time complexity and space complexity of the prediction approach should be flexible in a way that algorithm deployment is comfortable.

• Data granularity is the initial step to make flexibility of prediction model. Because the long term sampling data causes loss of effectiveness of the results. Selection of proper size of sample data could increase effectiveness of the model.

• Finding correlation among attributes and extracting needed features from sample data is the big challenge. Selection of proper pattern length will increase accuracy of the prediction.

The accurate prediction can be used to predict the appropriate amount of resource need to fulfill the demands. To achieve the accurate estimation of the future workload a reliable and precise prediction model need to employ. The main objective of this work is to predict the accurate CPU and memory utilization for different time intervals benefiting the cloud management for proper utilization of the available resources. The prediction model uses both statistical and machine learning approaches to give a comparative better quality prediction results with accuracy.

Also, in cloud computing the energy consumption and resource utilization are relate to each other. Mainly, resources with a low utilization rate still consume an unacceptable amount of energy compared to the energy consumption of a fully utilized or sufficiently loaded cloud computing. To increase resources utilization, task consolidation is an effective technique which greatly enabled by virtualization technologies, that facilitate the concurrent execution of several tasks and in turn reduce the energy consumption.

1.4 Related Work:

Statistical Models:

A better solution toward smart cloud management is to have a good knowledge about the current and future workload patterns of the applications, including their CPU, memory, disk, and network demands. With a sufficient knowledge about the workloads of the applications, the cloud can make smart management decisions proactively and avoid potential overhead at a future time. Workload prediction has been widely investigated over several decades and many techniques has been proposed in the literature to improve prediction accuracy. In **[4]** proposed a hybrid method that combines wavelet decomposition and autoregressive integrated moving average (ARIMA) to predict it at the next time interval. An ARIMA model is established for the statistical characteristics of the trend and components, respectively. In **[6]** they proposed an Advanced Model for Efficient Workload Prediction in the Cloud (AME-WPC), which combines statistical and learning methods, improves accuracy of workload prediction for cloud computing applications and can be dynamically adapted to a particular system. In **[7]** they consider a job-pool based approach, where the knowledge about the workloads of a

large pool of tasks is used to help predict the workloads of new tasks. They develop a clustering-based learning approach to realize the job-pool based concept. The pools of jobs are clustered based on their workloads, and a neuralnet is used to learn the characteristics of the workloads in each cluster. When a new job arrives, we use its initial workload pattern and submission parameters to find the cluster it belongs to. Then, the corresponding neuralnet is used to predict the workload of the new job far into the future. In **[8]** An Intelligent Regressive Ensemble Approach for Prediction (REAP) has been proposed which integrates feature selection and resource usage prediction techniques to achieve high performance. The effectiveness of proposed approach is evaluated in a real cloud environment by conducting a series of experiments. The experimental results show that the proposed approach outperforms the existing models by significantly improving the accuracy rate and reducing the execution time. In **[9]** they present an approach for predicting Cloud resource utilization on a per-task and per-resource level. For this, they apply machine learning-based prediction models. Based on extensive evaluation, they show that they can reduce the prediction error by 20% in a typical case, and improvements above 89% are among the best cases. Exponentially Weighted Moving Average (EWMA) algorithm, which deals with dynamic consolidation of VMs based on an analysis of historical data of the resource usage by VMs, is proposed in **[10]** to detect overloaded hosts. It improves the calculation of the upper threshold for host overloading and, as a result, improves the identification of the problem with VM migration.

Machine Learning Model:

In **[11]** a new approach for VM workload prediction based on deep learning was proposed. A deep learning prediction model was designed with a deep belief network (DBN) composed of multiple-layered restricted Boltzmann machines (RBMs) and a regression layer. The DBN is used to extract the high level features from all VMs workload data and the regression layer is used to predict the workload of the VMs in the future. In **[12]** investigates the trace logs of real datacenter workload arrival trend in order to characterize the resource request trend of users. Furthermore, anticipated resource request trends in terms of the requested level of CPU and memory resources are forecasted. A novel prediction model,

named LSTMtsw, is proposed based on recurrent neural networks of deep learning, which exploits historical data in combination with the current job arrival trend to predict the future resource request trend of users. the proposed LSTMtsw model integrates an effective mechanism for controlling the prediction effect to achieve energy conservation without degrading the service quality. In **[13]** the Time Series forecasting of CPU usage of machines in data centers using Long Short-Term Memory (LSTM) Network and evaluating it against the widely used and traditional autoregressive integrated moving average (ARIMA) models for forecasting. The final LSTM model had a forecasting error in the range of 17-23% compared to ARIMA model's 37-42%. The results clearly show that LSTM models performed more consistently due to their ability to learn non-linear data much better than ARIMA models. In **[14]** The SARIMA model outperformed the LSTM for the long term prediction task, but performed poorer on the short term task. Furthermore, the LSTM model was more robust, whereas the SARIMA model relied on the data meeting certain assumptions about seasonality. Research on developing tools and techniques have been investigated and implemented in computational grids such as a Bayesian model to determine short and long-term virtual resource requirement of the CPU / memory intensive applications on the basis of workload patterns at several data centers in the cloud during several time intervals**[15].** An Ensemble based workload prediction mechanism is proposed that is based on stack generalization to improve the prediction accuracy by 2% **[16].** The proposed model is compare with the individual and baseline prediction models. Root Mean Square Error (RMSE) used for the comparison with the baseline model. It has shown 6% and 17% reduction in RMSE in CPU usage and in Memory usage prediction respectively. A self-adaptive prediction method using ensemble model and subtractive-fuzzy clustering based fuzzy neural network (ESFCFNN) **[17].**

Recourse Utilization:

By analyzing the characters of user preferences and demands the architecture of the prediction model is constructed with some base predictors to compose the ensemble model. The fuzzy *c*-means combined with subtractive clustering algorithm by obtaining the number of fuzzy rules and the initial value of the premise and consequent parameters. Research on workload prediction for cloud computing elasticity automatic scaling mechanism, three models to predict the workload based on analyzing monitoring data**[18]**.

a) a Kalman fIlter model to predict the cloud workload

b) a novel pattern matching model to analyze and predict the workload;

c) a new trigger strategy for cloud computing elasticity automatic scaling mechanism to reduce the automatic scaling delay.

To minimize energy consumption of datacenters, it needs to consolidate cloud workloads into as few servers as possible and switch to sleep the redundant servers. Upon receiving a client request, the cloud scheduler creates a virtual machine (VM), allocates the requested resources (e.g., CPU and memory) to it, and assigns the VM to one of the cluster's physical machines (PMs). In current cloud resource allocation methods, the amount of resources specified by the client request remains reserved during the whole VM lifetime. In **[19]** some measurements on real Google traces and shows the one-day snapshot of this percentage. It observes that VMs only utilize about 35% and 55% of the requested CPU and memory resources.

As there is an increase demand in the use of cloud computing every day, the load on servers has increased to a great extent. Therefore the use of load balancing has been the focus of much research, due to the active and adaptable service that is offered by cloud computing. Some works on the concept of Green Resource Management Techniques (GRMT). To handle two components, a green based adaptive fault tolerance resource selection and green based virtualization of resource allocation. It is used to decreases the energy consumption, Carbon dioxide (CO2) emission and dynamic cost **[20].** GRMT is consists of green based adaptive fault tolerance resource selection, green based virtualization of resource allocation, green based match making and scheduling algorithms, green based cloud balancing and green based cloud brokering. A powerful algorithm for reducing the energy that is consumed in cloud computing known as the dynamic well-organized load balancing (DWOLB) algorithm for a well-organized use of resources **[21].** It includes virtual server management, virtual equipment, VMware server, operating system and server**.**

VM Consolidation (VMC) **[22]** is one such technique incorporated in CRMS to increase the energy-efficiency of Cloud. Hardware failure of existing Physical Machines (PMs) and addition of new PMs are continuous events in data center.

The resource requirement to accomplish the remaining tasks of existing service requests evolves with the course of time. Hence, as time progresses, remapping of remaining workload to currently available resources become inevitable to uphold the optimization of Cloud resource usage. The VM consolidation technique is applied to remap the remaining workloads to currently available resources which opt to migrate VMs into lesser number of active PMs, so that the PMs which would have no VM can be kept into sleep state. Since, energy consumption by the PM which is in sleep state is significantly lower than the energy consumption by the PM which is in active state, therefore, VM consolidation minimizes energy consumption of Cloud data center.

VM placement and VM migration both act as a backbone to the VM consolidation process**[23].** The issues such as heterogeneity and scalability of physical resources, volatile workloads and migration cost make the VM consolidation process difficult. The basic challenges for efficient VM consolidation can be summarized as follows:

· **Host under-load detection:** Deciding if a host is considered to be under-loaded so that all VMs should be migrated from it and the host should be switched to a low power mode (to minimize the number of active physical servers).

· **Host overload detection:** Deciding if a host is considered to be overloaded so that some VMs should be migrated from it to other active or reactivated hosts (to avoid violating the QoS requirements).

· **VM selection:** Selecting VMs to migrate from overloaded host.

· **VM live migration**: Performing VM migration process with minimal service downtime and resource consumption during migration process.

· **VM placement:** Virtual machine placement by placing VMs selected for migration on other active or reactivated hosts.

2. Chapter-II

2.1 Model Description:

Time series is a sequence of observations recorded at regular time intervals. Depending on the frequency of observations, a time series may typically be hourly, daily, weekly, monthly, quarterly and annual. Time series analysis involves understanding various aspects about the inherent nature of the series so that we are better informed to create meaningful and accurate forecasts. It involves developing models that best capture or describe an observed time series in order to understand the underlying causes. This often involves making assumptions about the form of the data and decomposing the time series into constitution components. More modern fields focus on the topic and refer to it as time series forecasting. Forecasting involves taking models fit on historical data and using them to predict future observations. The skill of a time series forecasting model is determined by its performance at predicting the future. Most time series data can be described by three components. And those are trend, seasonality and cyclic **[24]**.

Trend: A trend exists when there is a long-term increase or decrease in the data. It does not have to be linear. Sometimes we will refer to a trend as "changing direction", when it might go from an increasing trend to a decreasing trend.

Seasonality: A seasonal pattern occurs when a time series is affected by seasonal factors such as the time of the year or the day of the week. Seasonality is always of a fixed and known frequency.

Cyclic: A cycle occurs when the data exhibit rises and falls that are not of a fixed frequency. These fluctuations are usually due to economic conditions, and are often related to the "business cycle".

Many time series include trend, cycles and seasonality. When choosing a forecasting method, we will first need to identify the time series patterns in the data, and then choose a method that is able to capture the patterns properly.

A trend can be observed when there is an increasing or decreasing slope observed in the time series. Whereas seasonality is observed when there is a distinct repeated pattern observed between regular intervals due to seasonal factors. It could be

because of the month of the year, the day of the month, weekdays or even time of the day.

But it is not mandatory that all time series must have a trend and/or seasonality. A time series may not have a distinct trend but have seasonality. The opposite can also be true.

So, a time series may be imagined as a combination of the trend, seasonality and the error terms.

AR, MA, ARMA, and ARIMA models are the statistical models **[25]** that used to forecast the observation at (t+1) based on the historical data of previous time spots recorded for the same observation. Most of the time the trend & seasonality effects on data capture through this models.

2.1.1 Autoregressive (AR) model:

A regression model, such as linear regression, models an output value based on a linear combination of input values.

For example:

$$y_t = b_0 + b_1 * X_1$$

Where $\mathbf{y_t}$ is the prediction, $\mathbf{b_0}$ and $\mathbf{b_1}$ are coefficients found by optimizing the model on training data, and X is an input value.

This technique can be used on time series where input variables are taken as observations at previous time steps, called lag variables.

For example, we can predict the value for the next time step (t+1) given the observations at the last two time steps (t-1 and t-2). As a regression model, this would look as follows:

$$X_{(t+1)} = b_0 + b_1 * X_{(t-1)} + b_2 * X_{(t-2)}$$

Because the regression model uses data from the same input variable at previous time steps, it is referred to as an auto-regression (regression of self).

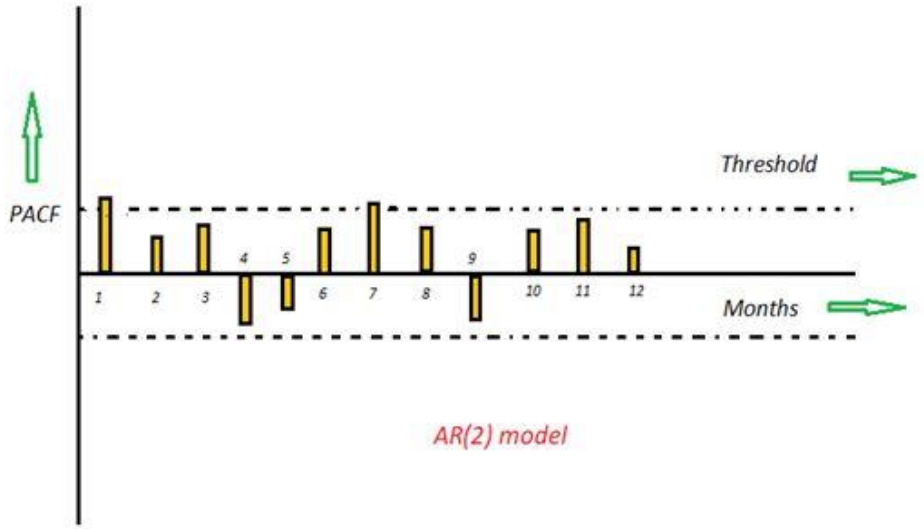


Fig. 5 Autoregressive Model with respect to PACF & Time

2.1.2 Moving Average (MA) Model:

Rather than using past values of the forecast variable in a regression, a moving average model uses past forecast errors in a regression-like model.

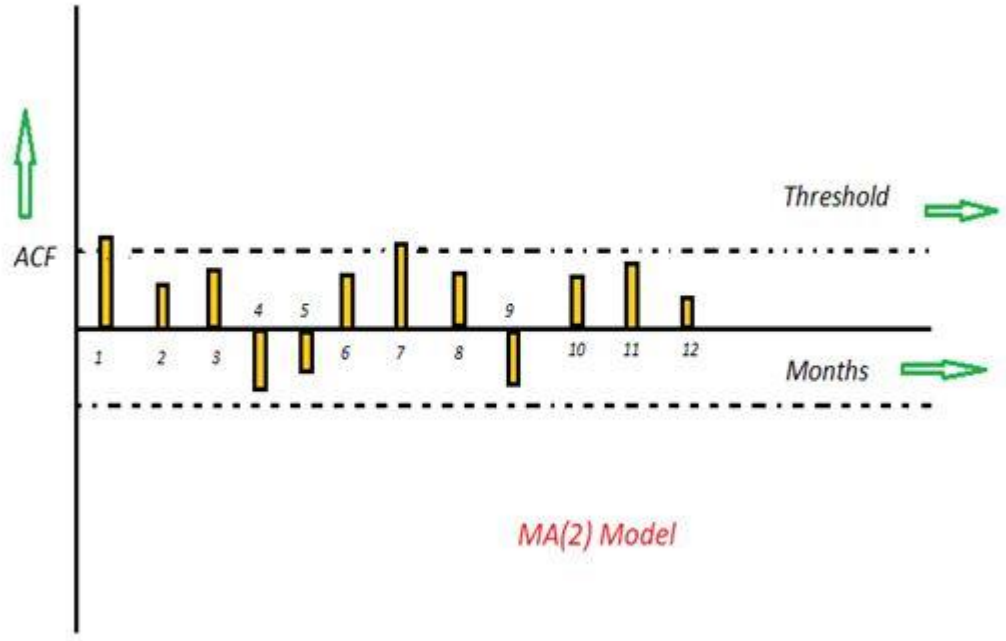


Fig. 6 Moving average Model with respect to ACF & Time

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q},$$

where ε_t is white noise. We refer to this as an **MA(q) model**, a moving average model of order q. each value of y_t can be thought of as a weighted moving average of the past few forecast errors. A moving average model is used for forecasting future values. Changing the parameters $\theta_1,\ldots,\theta_q$ results in different time series patterns. As with autoregressive models, the variance of the error term ε_t will only change the scale of the series, not the patterns.

2.1.3 Autoregressive Moving Average (ARMA) Model

ARMA is a model of forecasting in which the methods of autoregression (AR) analysis and moving average (MA) are both applied to time-series data that is well behaved. In ARMA it is assumed that the time series is stationary and when it fluctuates, it does so uniformly around a particular time.

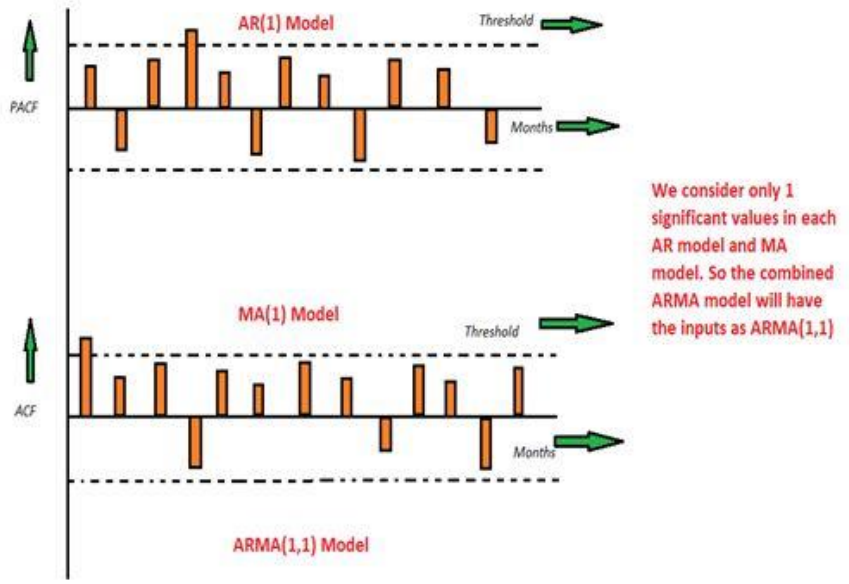


Fig. 7 ARMA Model with respect to PACF & ACF & Time

This is a model that is combined from the AR and MA models. In this model, the impact of previous lags along with the residuals is considered for forecasting the future values of the time series. Here β represents the coefficients of the AR model and α represents the coefficients of the MA model.

$$Yt = \beta_1 * y_{-1} + \alpha_1 * \varepsilon_{-1} + \beta_2 * y_{-2} + \alpha_2 * \varepsilon_{-2} + \beta_3 * y_{-3} + \alpha_3 * \varepsilon_{-3} + \ldots\ldots\ldots + \beta * y_{-} + \alpha * \varepsilon_{-}$$

Consider the above graphs where the MA and AR values are plotted with their respective significant values. Let's assume that we consider only 1 significant value from the AR model and likewise 1 significant value from the MA model. So the ARMA model will be obtained from the combined values of the other two models will be of the order of ARMA(1,1).

2.1.4 ARIMA (Auto-Regressive Integrated Moving Average) Model

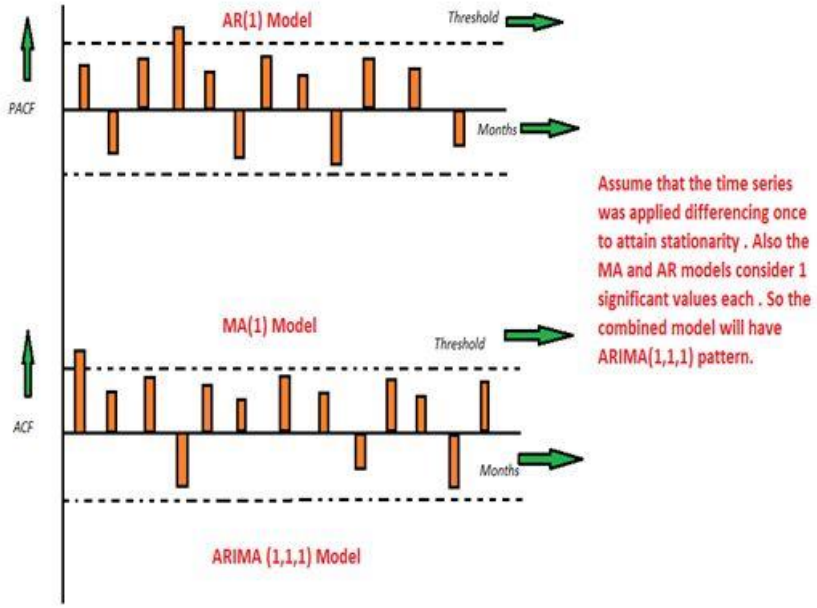


Fig. 8 ARIMA Model with respect to PCAF and ACF & Time

ARIMA is an acronym that stands for AutoRegressive Integrated Moving Average. It is a generalization of the simpler AutoRegressive Moving Average and adds the notion of integration.

This acronym is descriptive, capturing the key aspects of the model itself. Briefly, they are:

- **AR**: *Autoregression*. A model that uses the dependent relationship between an observation and some number of lagged observations.

- **I**: *Integrated*. The use of differencing of raw observations (e.g. subtracting an observation from an observation at the previous time step) in order to make the time series stationary.
- **MA**: *Moving Average*. A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.

Each of these components are explicitly specified in the model as a parameter. A standard notation is used of ARIMA(p,d,q) where the parameters are substituted with integer values to quickly indicate the specific ARIMA model being used.

The parameters of the ARIMA model are defined as follows:

- **p**: The number of lag observations included in the model, also called the lag order.
- **d**: The number of times that the raw observations are differenced, also called the degree of differencing.
- **q**: The size of the moving average window, also called the order of moving average.

A linear regression model is constructed including the specified number and type of terms, and the data is prepared by a degree of differencing in order to make it stationary, i.e. to remove trend and seasonal structures that negatively affect the regression model.

A value of 0 can be used for a parameter, which indicates to not use that element of the model. This way, the ARIMA model can be configured to perform the function of an ARMA model, and even a simple AR, I, or MA model.

Adopting an ARIMA model for a time series assumes that the underlying process that generated the observations is an ARIMA process. This may seem obvious, but helps to motivate the need to confirm the assumptions of the model in the raw observations and in the residual errors of forecasts from the model.

Consider the above graphs where the MA and AR values are plotted with their respective significant values. Let's assume that we consider only 1 significant value

from the AR model and likewise 1 significant value from the MA model. Also, the graph was initially non-stationary and by performing differencing operation once in order to convert into a stationary set. Hence the ARIMA model which will be obtained from the combined values of the other two models along with the Integral operator can be displayed as ARIMA(1,1,1).

A stationary time series is one whose properties do not depend on the time at which the series is observed. Thus, time series with trends, or with seasonality, are not stationary — the trend and seasonality will affect the value of the time series at different times. The cloud datasets are stationary in nature as there are no such trends or seasonality observed on it. So, the machine learning models like LSTM will works better on it to give the better prediction result than the statistical model.

2.1.5 EWMA(Exponentially weighted moving averages)

EWMA is a synonym for first-order exponential smoothing — or simple exponential smoothing. EWMA applies weights to the values of a time series. More weight is applied to more recent data points which make them more relevant for future forecasts. It's the most significant improvement over simple moving averages.

The only decision a user of the EWMA must make is the parameter alpha. The parameter decides how important the current observation is in the calculation of the EWMA. The higher the value of alpha, the more closely the EWMA tracks the original time series.

The EWMA's simple mathematical formulation described below:

$$\mathrm{EWMA}_t = \alpha * r_t + (1 - \alpha) * \mathrm{EWMA}_{t-1}$$

Where:

- **Alpha** = The weight decided by the user
- **r** = Value of the series in the current period

The EWMA is a recursive function, which means that the current observation is calculated using the previous observation. The EWMA's recursive property leads to the exponentially decaying weights as shown below:

$$EWMA_t = \alpha * r_t + (1 - \alpha) * EWMA_{t-1}$$

The above equation can be rewritten in terms of older weights, as shown below:

$$EWMA_t = \alpha * r_t + (1 - \alpha) * (\alpha * r_{t-1} + (1 - \alpha) * EWMA_{t-2})$$

It can be further expanded by going back another period:

$$EWMA_t = \alpha * r_t + (1 - \alpha) * \left(\alpha * r_{t-1} + (1 - \alpha) * (\alpha * r_{t-2} + (1 - \alpha) * EWMA_{t-3})\right)$$

The process continues until we reach the base term $EWMA_0$. The equation can be rearranged to show that the $EWMA_t$ is the weighted average of all the preceding observations, where the weight of the observation r_{t-k} is given by:

$$\alpha * (1 - a)^k$$

Since alpha is between 0 and 1, the weight becomes smaller as k becomes larger. In other words, as we go back further in history, the weight becomes smaller. The fact is illustrated in the chart below, which plots the weights of observation as k increases for different choices of the parameter alpha.

2.1.6 LSTM (Long Short Term Memory Network)

In deep learning there arises some situation, where we need to have the knowledge about the previous data in order to predict the next output. For example, in order to predict the next word of a sentence, we should know its previous words, in such scenarios our usual neuron won't help as it doesn't know the previous words. RNN (Recurrent neural network) comes into play in these kinds of situation. RNN is a kind of neuron cell, which has the ability to retain information about the sequence.

The hidden state of RNN passes information from a time-step to another time-step, hence remembering the sequence.

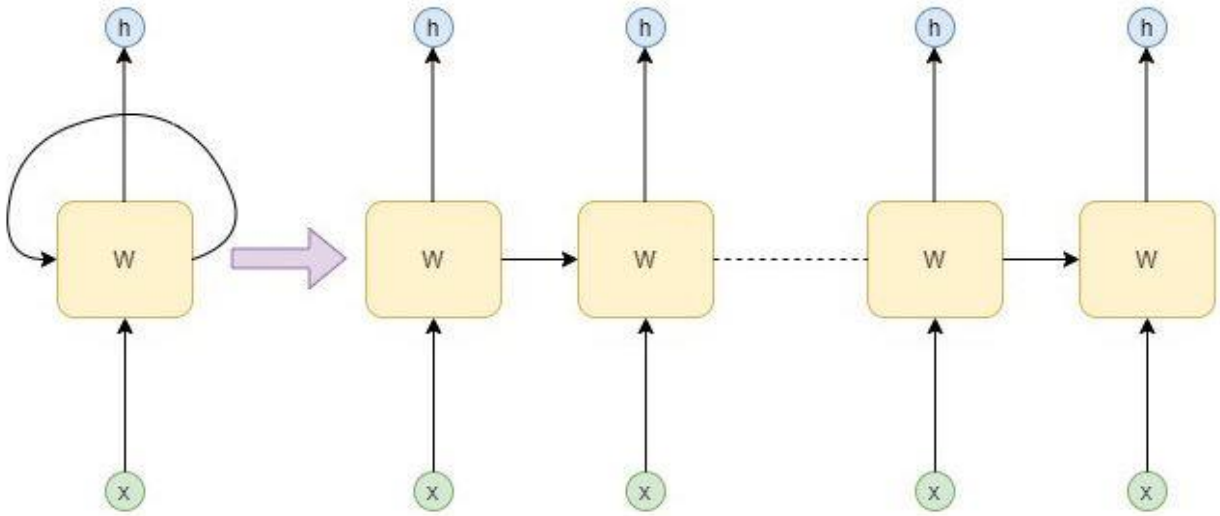


Fig. 9 Structure of RNN

The main advantage of RNN is that it can retain the information about the data that it processed in the past. When an input x is passed into the RNN cell it will be multiplied with a weight W and a bias b will be added. Then the calculated value will be non-linearized by an activation function like sigmoid, ReLu etc.

If we consider sigmoid function, for a range of input values, it will return an output within the range of (0,1). If we consider a time-step of 4 and the output of sigmoid being 0.3. At the end of 4th time step the output y of the RNN will be

$$Y=0.3*0.3*0.3*0.3*x + b .$$

x is being multiplied with a very small number which can be approximated to zero, which will result in zero gradient during the back-propagation, as a result, the weight won't be optimized. This issue is known as the vanishing gradient problem.

This will result in, initial layers not being optimized/trained. Since the latter layers depends on the low level features captured by the initial layer, overall performance of the model will be degraded. If the activation function is ReLu, then in case of negative input it will output zero, which then causes the same issue, even if we use Leaky-ReLu, the outputs for negative values are too small to overcome the

vanishing gradient problem. So changing activation functions was not enough to deal with vanishing gradient problem.

The main reason for using RNN is to retain the information gained from previous data, but if we are facing vanishing gradient issues the amount of data that can be stored in the memory is very much limited.

LSTM **[26]** is introduced as a solution to this problem. Long Short Term Memory is capable of storing processed information about the longer sequence of data and can solve the vanishing gradient problem.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

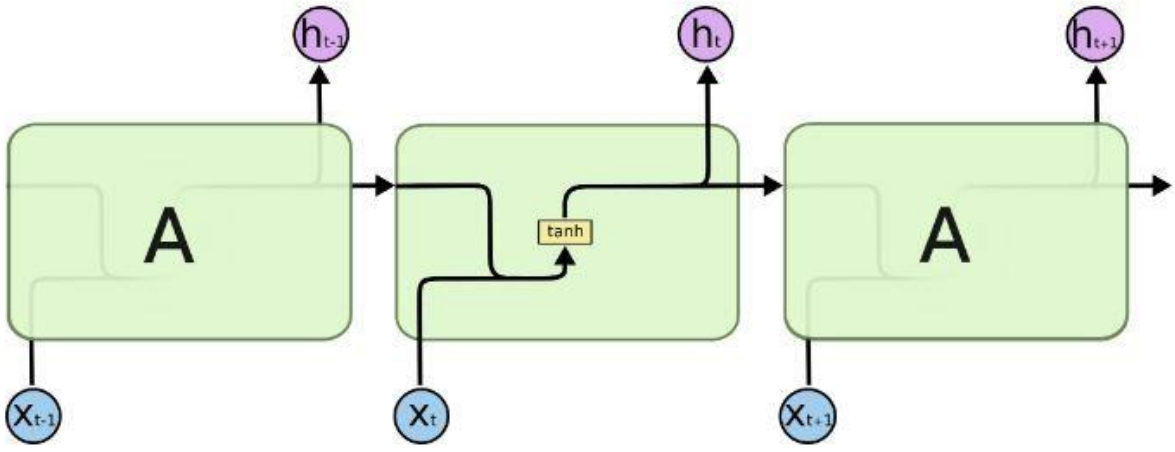


Fig. 10 The repeating module in a standard RNN contains a single layer

STMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.

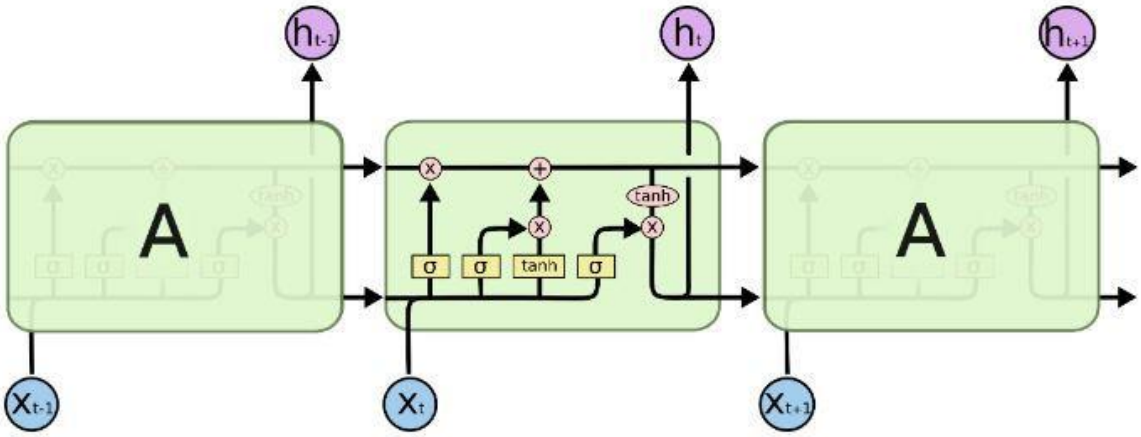


Fig. 10 The repeating module in an LSTM contains four interacting layers

In the above diagram, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent point wise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denote its content being copied and the copies going to different locations.

The key to LSTMs is the cell state, the horizontal line running through the top of the Fig. 11.

The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unchanged.

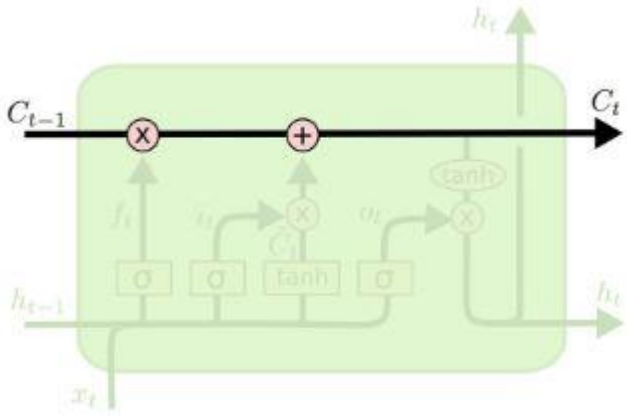


Fig 11. LSTMs is the cell state

The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.

Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a point wise multiplication operation.

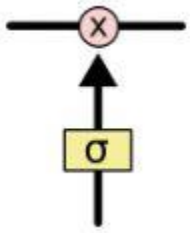
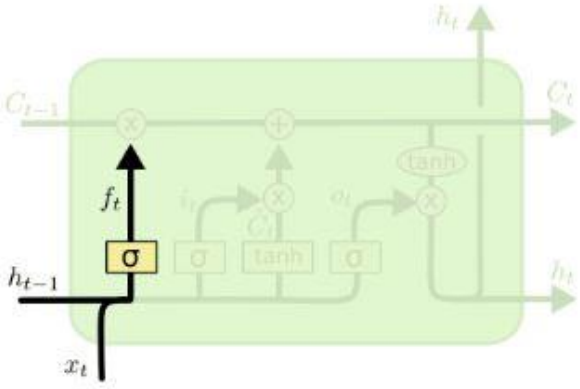


Fig. 12 The sigmoid layer

An LSTM has three of these gates, to protect and control the cell state.

The first step in the LSTM is to decide what information we're going to throw away from the cell state. This decision is made by a sigmoid layer called the "forget gate layer." It looks at h_{t-1} and x_t, and outputs a number between 00 and 11 for each number in the cell state C_{t-1}. A 11 represents "completely keep this" while a 00 represents "completely get rid of this."

Considering the example of the language model trying to predict the next word based on all the previous ones. In such a problem, the cell state might include the gender of the present subject, so that the correct pronouns can be used. When it see a new subject, it want to forget the gender of the old subject.

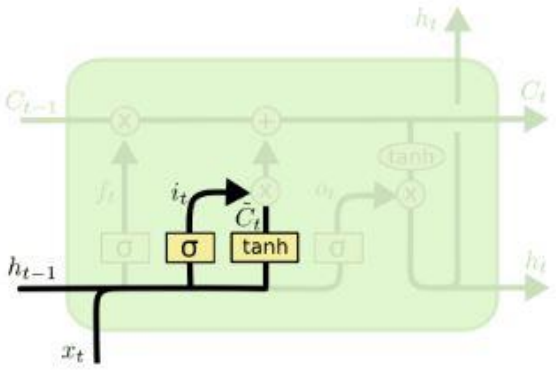


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Fig.13 Calculation of f_t

The next step is to decide what new information it's going to store in the cell state. This has two parts. First, a sigmoid layer called the "input gate layer" decides which values it'll update. Next, a tanh layer creates a vector of new candidate values, C_t, that could be added to the state. In the next step, it 'll combine these two to create an update to the state.

Considering our language model, it wants to add the gender of the new subject to the cell state, to replace the old one forgetting.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

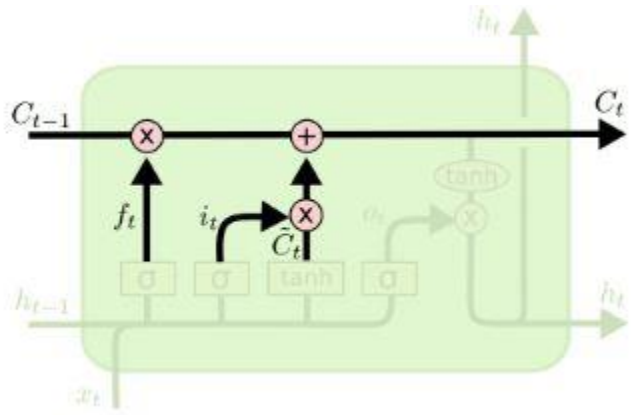
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Fig.14 Calculation of C_t & i_t

It's now time to update the old cell state, C_{t-1}, into the new cell state C_t. The previous steps already decided what to do, we just need to actually do it.

It multiply the old state by f_t, forgetting the things it decided to forget earlier. Then it add $*C_t$. This is the new candidate values, scaled by how much we decided to update each state value.

In the case of the language model, this is where it actually drop the information about the old subject's gender and add the new information, as decided in the previous steps.

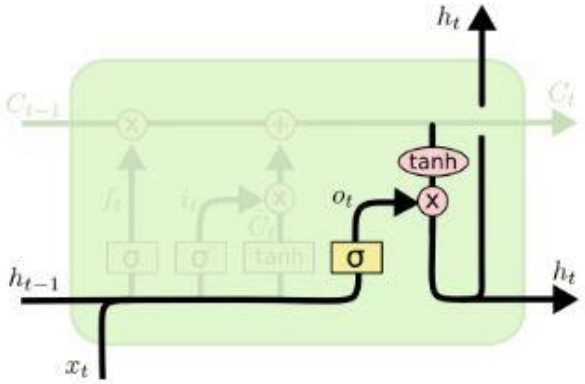


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Fig.15 Re calculation of C_t

Finally, it need to decide what going to output. This output will be based on our cell state, but will be a filtered version. First, it run a sigmoid layer which decides what parts of the cell state we're going to output. Then, it put the cell state through tanhtanh (to push the values to be between −1 and 1) and multiply by the output of the sigmoid gate, so that we only output the parts we decided to.

For the language model example, since just saw a subject, might want to output information relevant to a verb, in case that's what is coming next. For example, might output whether the subject is singular or plural, so that we know what form a verb should be conjugated into if that's what follows next.



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Fig.16 Calculation of o_t & h_t

3. Chapter-III

3.1 Experimental Setup:

3.1.1 Dataset Description:

A publicly available workload trace called fastStorage is used to predict the CPU & Memory utilization in future of the virtual machines for better utilization of data centre resources on a large-scale virtualized data centre. The trace was gathered from a typical data centre run by BitBrains, a company that specializes in managed hosting and business computation for large businesses. The trace is typical for business-critical workloads, which are made up of program that frequently run on the same virtual machines for extended periods of time, typically over several months. In one operating month, the trace collects performance data for 1250 virtual machines (VMs) across different resource types (e.g CPU, memory, network, and disc), sampled every 5 minutes.

Trace source	Properties	Mean	Min	Q1	Median	Q3	Max	CoV	SDev
BitBrains	Memory requested [GB]	10.7	0	1.27	3.98	15.59	511	2.8	29.3
BitBrains	Memory usage [GB]	0.6	0	0.03	0.1	0.29	384	3	1.8
BitBrains	CPU requested [GHz]	8.9	2.4	2.93	5.2	10.4	86	1.3	11.1
BitBrains	CPU usage [GHz]	1.4	0	0.02	0.08	0.2	64	3.3	4.4

Table 1. Statistics of requested & used resources of BitBrains for CPU & Memory

For the purposes of this experiment, only the metrics for CPU and memory have been chosen, including: provisioned CPU capacity, actual CPU usage, provisioned memory capacity, and actual memory usage. The average over the sampling interval represents each and every value of the metrics. The difference between provisioned and actual usage of the CPU and memory is captured through further analysis of the trace. The table in this paragraph displays some metrics statistics. On averaging the values of several VMs divide the dataset into two parts referred as the training and the testing set.

3.1.2 Workload Prediction:

A variety of well-known Statistical models have been used for prediction along which a Machine Learning model for the VMs are selected based on s capacity to produce results which a minimum amount of error. The Statistical models includes Autoregressive (AR), Autoregressive Integrated Moving Average(ARIMA) & EWMA(Exponentially weighted moving averages) also with the ML model LSTM (Long Short Term Memory Network). The VM's historical resource usage average data, for CPU & Memory capacity provisioned and usage has been taken. Then the data split for training to fit into the model for future prediction. The test data also used to compare with the predicted values to calculate the error rate. The function & the parameters used in each model described below.

AR Model:

An auto-regression model is a linear regression model that uses lagged variables as input variables. By using the statsmodels library which provides an autoregression model where specify an appropriate lag value and trains a linear regression model. is provided in the AutoReg class. It can use this model by first creating the model AutoReg() and then calling fit() to train it on the dataset. This returns an AutoRegResults object. Once fit, it can use the model to make a prediction by calling the predict() function for a number of observations in the future. By changing the value of "lag" from 3 to 7 get different error percentage along with the graphical representation. The best graphical representation shown over here in Fig. 8 with "lag" value as 3.

ARIMA Model:

The statistical models library provides the capability to fit an ARIMA model. An ARIMA model can be created using the statistical models library as follows:

1. Define the model by calling ARIMA() and passing in the *p*, *d*, and *q* parameters.
2. The model is prepared on the training data by calling the fit() function.
3. Predictions can be made by calling the predict() function and specifying the index of the time or times to be predicted.

First, it fit an ARIMA(2,1,2) model. This sets the lag value to 2 for autoregression, uses a difference order of 1 to make the time series stationary, and uses a moving average model of 2.

EWMA Model:

It can use the *ewm()* function in Pandas to calculate exponentially weighted moving averages. By passing the smoothing value directly through alpha or make easier with the span parameter. Both should sound familiar by now. Then by calling *fit()* to train on the dataset. Once fit, it can use the model to make a prediction by calling the *forecast( )* function for a number of observations in the future. By changing the value of the parameter "span" from 3 to 20 it will vary the value of the parameter "alpha". It is observed that with the different value of "alpha" the predicted values in the graph moving up & down and also changes the error percentage. The best value is taken over here & shown in the Fig. 19 & Fig. 23.

LSTM Model:

Recurrent neural networks (RNN) include the Long Short-Term Memory network (LSTM). This type of network has the advantage of learning and remembering over extended periods of time without requiring input from pre-determined window-lagged observations.

An LSTM layer often keeps state between the data in a single batch. A fixed-sized number of rows from the training dataset make up a batch of data, which indicates how many patterns should be processed before the network's weights are updated. We must make the LSTM stateful since between batches, state is by default cleared in the LSTM layer. By using the *reset_states()* function, this provides us precise control over when the state of the LSTM layer is cleared.

The LSTM layer expects input to be in a matrix with the dimensions: [*samples, time steps, features*].

- **Samples**: These are independent observations from the domain, typically rows of data.

- **Time steps**: These are separate time steps of a given variable for a given observation.
- **Features**: These are separate measures observed at the time of observation.

Frequently, there are fewer samples in a batch than there are overall. It determines how rapidly the network learns the material, coupled with the number of epochs.

The number of neurons, also known as the number of memory runs or blocks, is the one important element in defining the LSTM layer. A number between 1 and 5 should be adequate because this is a fairly straightforward task.

To anticipate the CPU or Memory utilization in the upcoming time step, the network needs a single neuron in the output layer with linear activation.

A backend mathematics library, such as TensorFlow or Theano, must be used to compile the network into an effective symbolic representation once it has been specified.

It must define a loss function and optimization strategy when building the network. Since it closely resembles the RMSE in which we are interested and the effective ADAM optimization approach, we will choose "mean squared error" as the loss function.

It can be fitted to the training data after being compiled. must manage when the internal state is reset because the network is stateful. Therefore, over the appropriate number of epochs, manually need to manage the training process one epoch at a time.

At the conclusion of each epoch, the network by default outputs a lot of debug information regarding the model's skill and learning progress. This can be turned off by setting the "verbose" argument's level to "0," after which it restores the internal state to prepare for the subsequent training iteration.

It can define a function called *fit_lstm()* that trains and returns an LSTM model. As arguments, it takes the training dataset in a supervised learning format, a batch size, a number of epochs, and a number of neurons.

Here, the *batch_size* must be set to 1. This is because must be a factor of the size of the training and test datasets also want to make one-step forecasts on the test data, is another constraint on the *predict()* method.

3.1.3 Performance metrics

To predict the resource need for CPU & Memory utilization and analyzing the performance of the Statistical and ML Models, one metrics used over here. Since VM's resource usage prediction using ML methods is a regression problem, so the used method is the Root-Mean Square Error (RMSE) and the Mean Absolute Error (MAE) to evaluate the performance of the prediction module. The RMSE shows how our module deviates from the actual value. The formula for calculating the RMSE & MAE given below:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{t=1}^{n} (a_t - p_t)^2}{n}}$$

$$\mathrm{MAE} = \frac{1}{n} \sum_{t=1}^{n} |a_t - p_t|$$

where a_t is the actual resource consumption and p_t is the estimated resource consumption at time interval t, and n is the number of performed estimations.

After trained the statistical & machine learning model with training dataset then the predicted values of the future resources were observed. Then to calculate the error percentage of each models between the predicted & test dataset need to use the RMSE.

3.2 Result:

Here, the results are demonstrate as per the findings and assess the performance of the statistical & ML model.

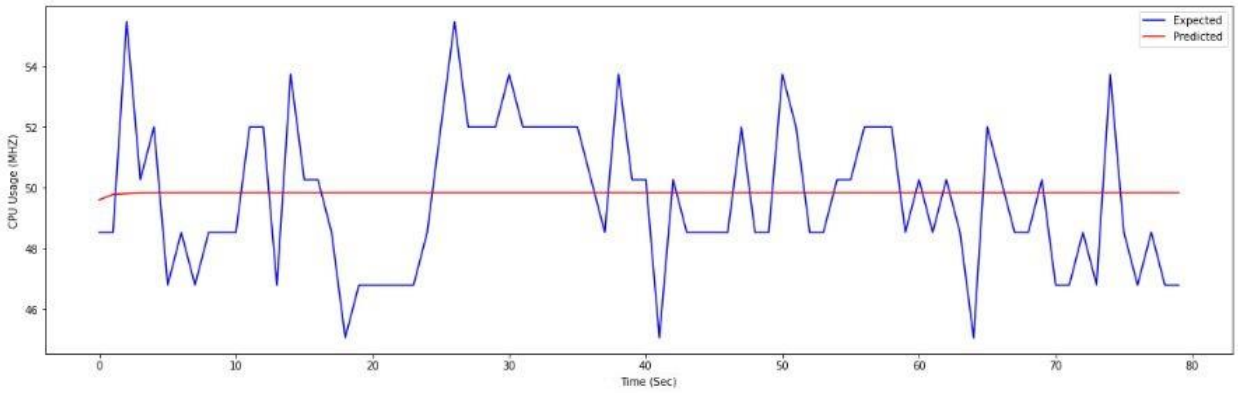


Fig. 17. CPU utilization for AR Model

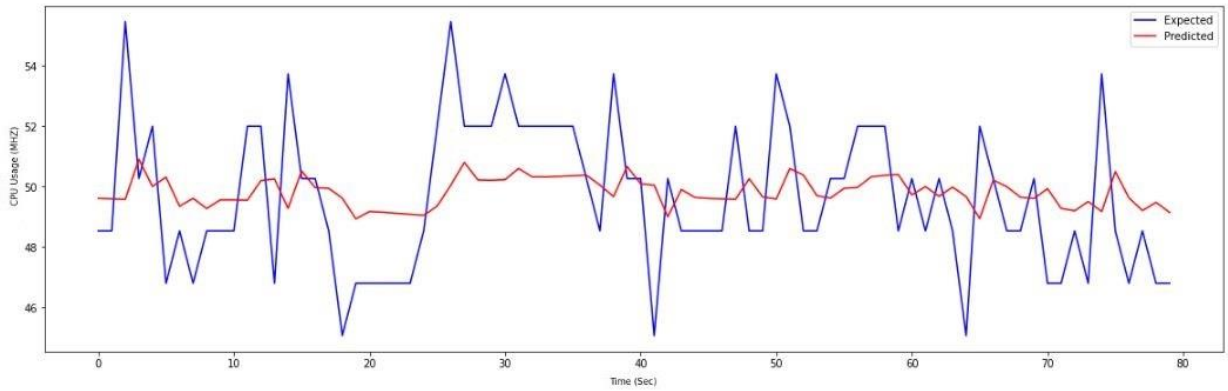


Fig. 18. CPU utilization for ARIMA Model

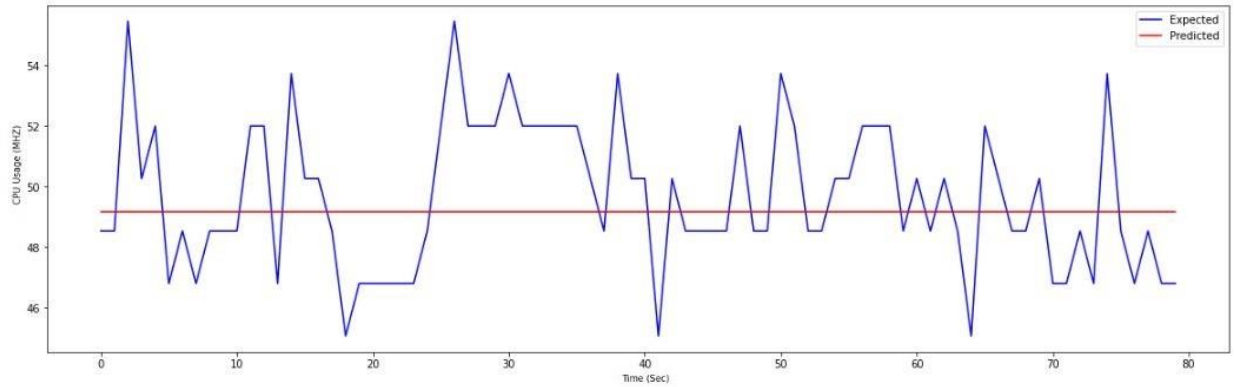


Fig. 19. CPU utilization for EWMA Model

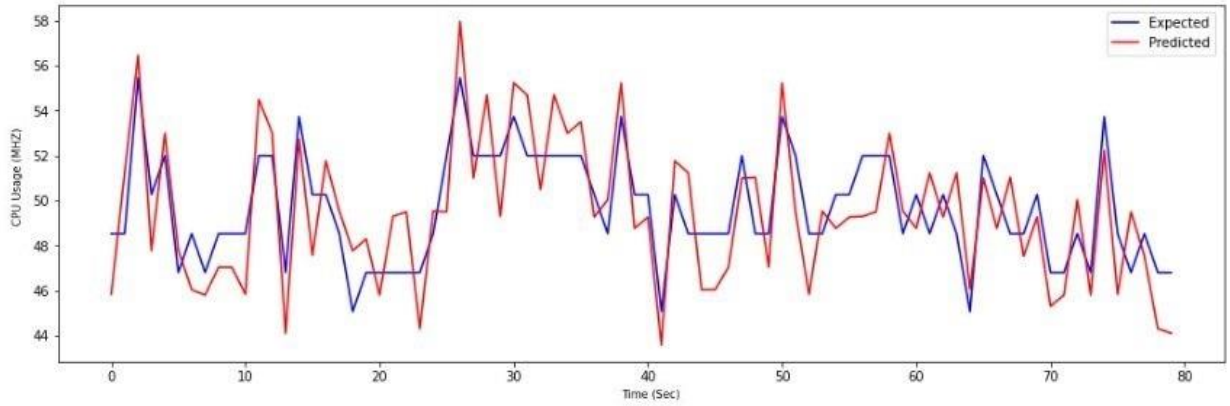


Fig. 20. CPU utilization for LSTM Model

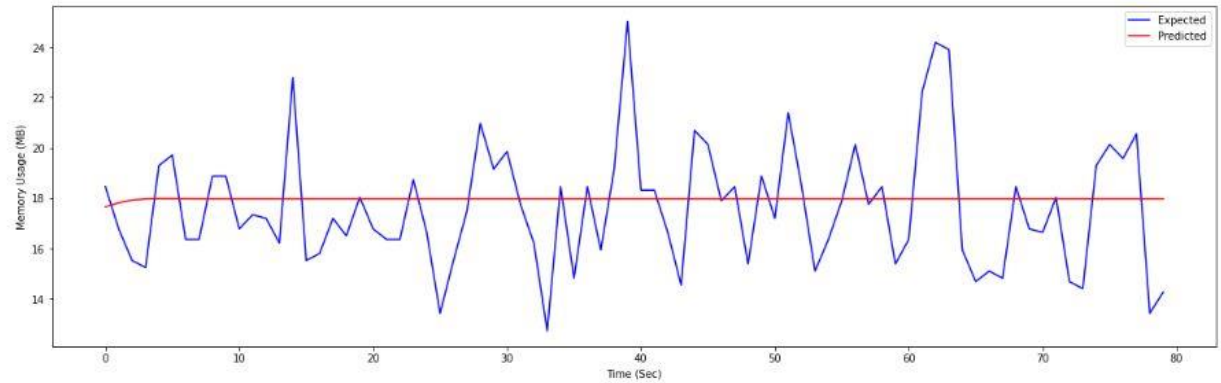


Fig. 21. Memory utilization for AR Model

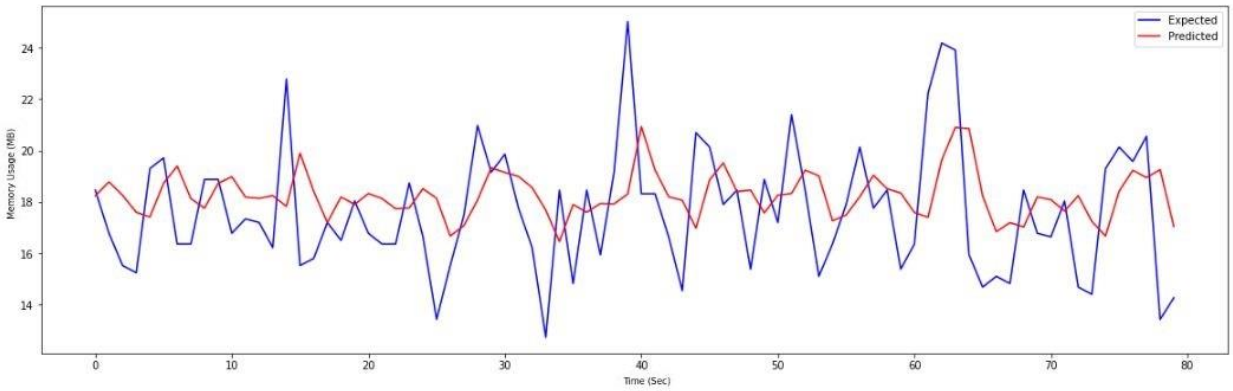


Fig. 22. Memory utilization for ARIMA Model

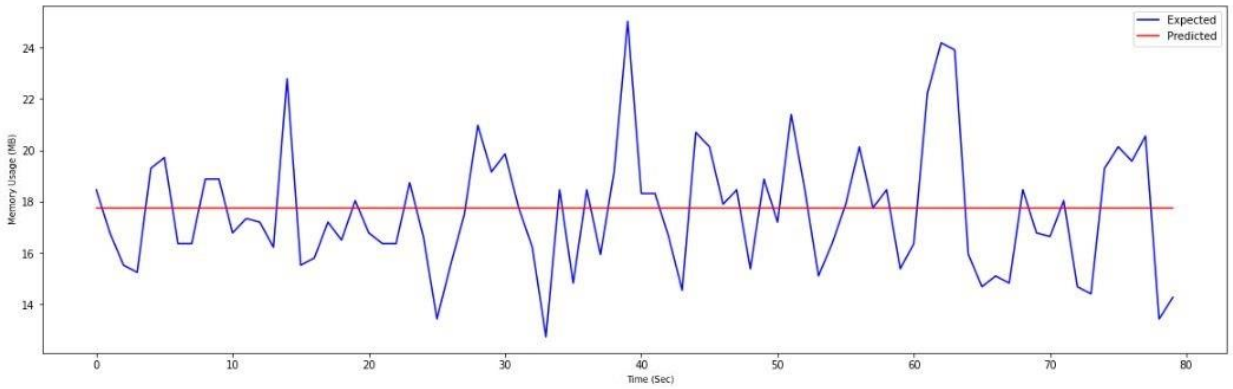


Fig. 23. Memory utilization for EWMA Model

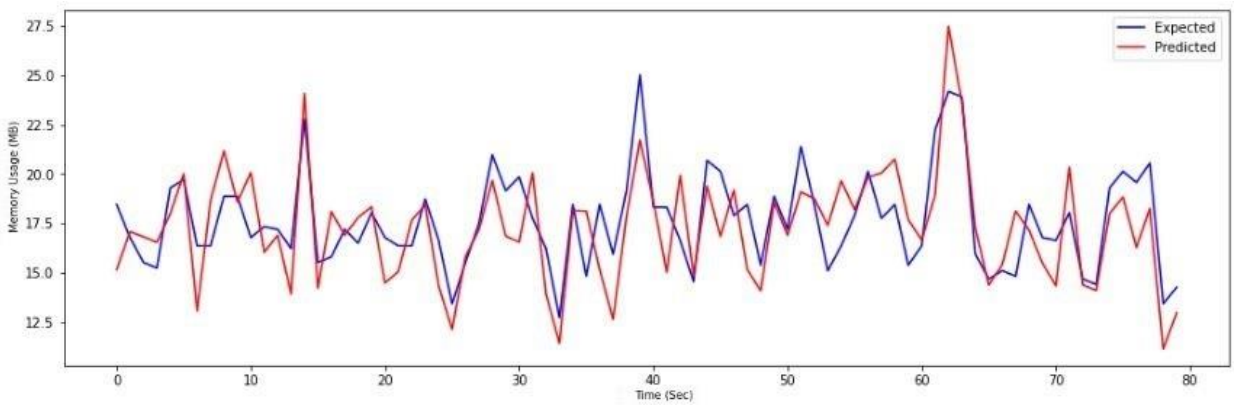


Fig. 24. Memory utilization for LSTM Model

Given below the error percentage result (i.e RMSE & MAE value) observed for each statistical model and the machine learning model.

RMSE value		
Model	**CPU Usage**	**Memory Usage**
AR	2.412	2.471
ARIMA	2.324	2.537
EWMA	2.457	2.451
LSTM	1.934	1.925

Table 2. RMSE values of CPU & Memory Usage

MAE value		
Model	**CPU Usage**	**Memory Usage**
AR	2.061	1.964
ARIMA	1.922	2.079
EWMA	2.025	1.939
LSTM	1.798	1.638

Table 3. MAE values of CPU & Memory Usage

3.3 Discussion:

From the above graphs & RMSE values of CPU utilization on different statistical model and the machine learning model is observed that the percentage of error rate varying in different statistical model in compare with AR, ARIMA & EWMA. ARIMA gives a good result compare with AR and EWMA. On the other hand the ML model LSTM gives better result than all the statistical model. From the graphs its observed that the machine learning model i.e LSTM is capable to capture the fluctuation with respect to the test values. Here the drawbacks of the statistical model are observed for the AR, ARIMA & EWMA and the LSTM works well.

4. Chapter –IV

4.1 Conclusion & Future work

In a cloud environment, managing resources profitably is difficult because of the high energy consumption and excessive operational costs. Additionally, PMs in a virtualized data centre experience constant resource fluctuation, which makes it more challenging to allocate resources efficiently. Resource utilization is one of the most important challenges for the Cloud provider. Future workload prediction is one of the ways to improve resource utilization. By using the predicted workload of the virtual machine in the cloud data centers can efficiently utilize the resources & make energy efficient.

Future research could go in a number of different directions as a result of this work. By predicting the workload of the VMs several dynamic VM consolidation techniques on the predicted values can be applied to see the effectiveness of the approach in terms of SLA violation and migration cost. Also placing of VMs on PM could be effective to reduce downtime and maintain QoS in accordance with SLA.

5. References:

[1] https://cloudcomputinggate.com/cloud-deployment-models/

[2] https://kinsta.com/blog/types-of-cloud-computing/

[3] Liu, Jinwei, Haiying Shen, and Liuhua Chen. "CORP: Cooperative opportunistic resource provisioning for short-lived jobs in cloud systems." In 2016 IEEE International Conference on Cluster Computing (CLUSTER), pp. 90-99. IEEE, 2016.

[4] Bi, Jing, Libo Zhang, Haao Yuan, and MengChu Zhou. "Hybrid task prediction based on wavelet decomposion and ARIMA model in cloud data center." In 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), pp. 1-6. IEEE, 2018.

[5] Amiri, Maryam, and Leyli Mohammad-Khanli. "Survey on prediction models of applications for resources provisioning in cloud." Journal of Network and Computer Applications 82 (2017): pp. 93-113.

[6] Cetinski, Katja, and Matjaz B. Juric. "AME-WPC: Advanced model for efficient workload prediction in the cloud." Journal of Network and Computer Applications 55 (2015): pp. 191-201.

[7] Yu, Yongjia, Vasu Jindal, Farokh Bastani, Fang Li, and I-Ling Yen. "Improving the smartness of cloud management via machine learning based workload prediction." IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), vol. 2, pp. 38-44. IEEE, 2018.

[8] Kaur, Gurleen, Anju Bala, and Inderveer Chana. "An intelligent regressive ensemble approach for predicting resource usage in cloud computing." Journal of Parallel and Distributed Computing 123 (2019): pp. 1-12.

[9] Borkowski, Michael, Stefan Schulte, and Christoph Hochreiner. "Predicting cloud resource utilization." In Proceedings of the 9th International Conference on Utily and Cloud Computing, pp. 37-42. 2016.

[10] Jen-Hsiang Chen, Shin-Li Lu, “Host Overloading Detection based on EWMA Algorhm in Cloud Computing Environment” IEEE 15th International Conference on e-Business Engineering (ICEBE) 2018

[11] Qiu, Feng, Bin Zhang, and Jun Guo. "A deep learning approach for VM workload prediction in the cloud." 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), pp. 319-324. IEEE, 2016.

[12] Wang, Hengjian, John Pannereselvam, Lu Liu, Yao Lu, Xiaojun Zhai, and Haider Ali. "Cloud workload analytics for real-time prediction of user request patterns." IEEE 20th International Conference on High Performance Computing and Communications; pp. 1677-1684. IEEE, 2018.

[13] Janardhanan, Deepak, and Enda Barrett. "CPU workload forecasting of machines in data centers using LSTM recurrent neural networks and ARIMA

models." In 2017 12th International Conference for Internet Technology and Secured Transactions (ICST), pp. 55-60. IEEE, 2017.

[14] Nashold, Langston, and Rayan Krishnan. "Using LSTM and SARIMA Models to Forecast Cluster CPU Usage." arXiv preprint arXiv:2007.08092, 2020.

[15] Gopal Kirshna Shyam, Sunilkumar S. Manvi, "Virtual Resource Prediction in Cloud Environment: A Bayesian Approach", Journal of Network and Computer Applications, 2016

[16] Tajwar Mehmood, Dr. Seemab Latif, Dr. Sheheryaar Malik, "Prediction Of Cloud Computing Resource Utilization", IEEE 2018

[17] Zhijia Chen, Yuanchang Zhu, Yanqiang Di, and Shaochong Feng, "Self-Adaptive Prediction of Cloud Resource Demands Using Ensemble Model and Subtractive-Fuzzy Clustering Based Fuzzy Neural Network", 2015

[18] Yazhou Hu, Bo Deng, Fuyang Peng, Dongxia Wang, "Workload Prediction for Cloud Computing Elasticy Mechanism", International Conference on Cloud Computing and Big Data Analysis IEEE 2018

[19] Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizan and Ammar Rayes, "Efficient Datacenter Resource Utilization Through Cloud Resource Overcommment", INFOCOM Workshop on Mobile Cloud and Virtualization, IEEE 2015

[20] Surendran.R, Tamilvizhi.T, “How to Improve the Resource Utilization in Cloud Data Center?” International Conference on Innovation and Intelligence for Informatics, IEEE 2018

[21] M. Vanha, P. Marikkannu, "Effective resource utilization in cloud environment through a dynamic well-organized load balancing algorhm for virtual machines", Computers and Electrical Engineering, 2016

[22] Md An Khan, Andrew Paplinski, Abdul Malik Khan, Manzur Murshed, and Rajkumar Buyya, "Dynamic Virtual Machine Consolidation Algorhms for Energy-Efficient Cloud Resource Management: A Review", Springer International Publishing, 2018

[23] Amany Abdelsamea1, Elsayed E. Hemayed, Hesham Eldeeb, and Hanan Elazhary, "Virtual Machine Consolidation Challenges: A Review", Innovative Space of Scientific Research Journals, 2014

[24] https://otexts.com/fpp2/tspatterns.html

[25] https://towardsdatascience.com/time-series-models-d9266f8ac7b0

[26] https://colah.ghub.io/posts/2015-08-Understanding-LSTMs/