Auto Scaling of Cloud Resources using Time Series and Machine Learning Prediction

A thesis submitted for the Degree of

Master of Science (Research)

By

Sivasankari Bhagavathiperumal

In

School of Computer Science

UNIVERSITY OF TECHNOLOGY SYDNEY

AUSTRALIA

MARCH 2020

CERTIFICATE OF ORIGINAL AUTHORSHIP

Date: March 2020

Author: Sivasankari Bhagavathiperumal

Title: Auto Scaling of Cloud Resources Using Time Series and Machine Learning

Degree: Master of Science (Research) in Computing Sciences

I, Sivasankari Bhagavathiperumal declare that this thesis, is submitted in fulfilment of the requirements for the award of Master of Science (Research) in computing Sciences, in the School of Computer Science at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This research is supported by the Australian Government Research Training Program.

Signature: Production Note:
Signature removed prior to publication.

Date: 02/10/2020

Acknowledgments

My first acknowledgement is to God for being my source of strength. I would like to thank my supervisor Dr. Madhu Goyal for the opportunity to work with her on the thesis, for all the guidance she has given me along the way. I thank the University of Technology Sydney for providing me a wonderful education environment and resources that made this work possible. I would also thank the experts who were involved in validating my project. Finally, I thank my parents, husband and kids for supporting me in my years of study. Thank you all.

Table of Contents

Table of Contents

List of Tables

List of Figures

Abstract

Cloud platforms are becoming very popular as a means to host business applications, and most businesses are starting to use the cloud platform in order to reduce costs. A cloud platform is a shared resource that enables businesses to share the services Software as a service (SAAS), Infrastructure as a service (IAAS) or Anything as a service (XAAS), which are required to develop and deploy any business application. These cloud services are provided as virtual machines (VM) that can handle the end user's requirements. The cloud providers must ensure efficient resource handling mechanisms for different time intervals to avoid wastage of resources. Auto-scaling mechanisms would take care of using these resources appropriately along with providing an excellent quality of service. Therefore, a process to predict the workload is required so that the cloud servers can handle the end user's request and provide the required resources as Virtual Machines (VM) disruptively, so that the businesses using the cloud service only pay for the service they use, thus increasing the popularity of cloud computing. The workload consists of the application programs running on the machine and the users connected to and communicating with the computer's applications. The computing resources are released based on the workload identified using the resource provisioning techniques, which are models used for enabling convenient on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, applications and services).

In order to provide these resources dynamically without any interruption, an auto-scaling system is required to both manage the load and balance the service. The application should be capable of handling the load as it comes in and to ensure the cloud resources are not supplied in abundance when there are fewer requests coming to the server. This thesis introduces a framework to identify the current workload and predict the future load to the server. The results show the actual workload to the server and the level of requests expected in the future. But to fully appreciate the flexibility of identifying the future load in advance, it was important to identify a method that could in turn provide guidance with adjusting the resources supplied reducing the cost. Also, this would ensure the service level agreement (SLA) was met. The framework identified in this thesis carefully monitors the inputs to the server and analyses the load. The researches also applied deep learning techniques to predict the future load to the server, which produced a result with less root mean square error (RMSE) compared the other techniques. This thesis investigates five different methods to identify the future workload: Naïve, ARIMA, Linear regression, Exponential smoothing and Forward propagation. The logs of two datasets were analysed to predict the future resource for different time intervals along with computing the error between the predicted and actual value.

Chapter 1

1. Introduction

Cloud computing is a popular technology that provides the distributed infrastructure or services that include environments to host vendor applications, network resources, build and deploy applications. Instead of setting up their own infrastructure, users of cloud computing, such as online stores and webmasters, tend to prefer the cloud services, such as Amazon Elastic Compute Cloud (Amazon EC2), Microsoft Azure, who offer resources in terms of Virtual Machines (VM) (Aslanpour, Ghobaei-Arani & Nadjaran Toosi 2017). Scaling, which allocates resources dynamically based on the volume of the request to the server, is one of the most prominent features of these services. The cloud servers offer two types of scaling, namely horizontal scaling and vertical scaling. Horizontal scaling is connecting multiple servers at the same time to do the same work, increasing the speed or availability of the logical units. Vertical scaling is the ability to raise the power of the server, for example by increasing the RAM of the computer, to improve the speed and performance of the computer (Liu et al. 2014).

It is the elastic nature of cloud computing, where in the cloud users can pay based on what they use, that has led to its success. Cloud providers use the auto-scaling mechanism to ensure that there is no over-provisioning or under provisioning of resources. In cloud settings, resources can be dynamically provisioned with the aim of reducing the cost. However, these resources are provided during the run time in most of the scenarios. Providing these resources during run time is more cost effective than deciding the resources in advance, which is the biggest challenge faced by the cloud service providers. Determining these resources depends on the workload, which can be identified in advance using various approaches. It is important to predict the future resource requirements so that unstable loads can be dealt with and to offer cloud services without any limitations (i.e., resources are assigned automatically without any interruptions).

Moreover, predicting future resource is also required to maintain low cost by turning off the supply when the service is not being used by users, thus ensuring the service level agreement (SLA) is not violated. Auto scaling is the provisioning of resources through virtual machines, ensuring the quality of service in accordance with the Service Level Agreements (SLA). For example, consider a situation where X number of machines are serving N number of customers during a peak period at a Stock Exchange. Suddenly there is an unanticipated number of customers using the same application which increases the number of nodes. As a result, the performance of the application slows down, violating the SLA. At this time an efficient auto-scaling system would be required to manage the load and balances the service (Roy, Dubey & Gokhale 2011). Moreover, auto scaling ensures that the required sources of services are supplied seamlessly when the demand is high and the supply reduced when the demand decreases. The automated solution to this horizontal and vertical scaling would benefit both the cloud providers and the users of this cloud services, due to the intelligent and cost effective utilization of resource and the improved performance. However, identifying the required resources would be challenging, as the demand fluctuates from time to time (Roy, Dubey & Gokhale 2011). Therefore, it is important to build the frequently occurring events into the system so that the cloud systems predict the requirements of the users (Ghobaei-Arani, Jabbehdari & Pourmina 2016).

The cost of the cloud service is reduced if fewer resources are released, minimising any negative impact during peak hours of use (Yang et al. 2014). As shown in the figure below (Fig.1) (Aslanpour, Ghobaei-Arani & Nadjaran Toosi 2017), the auto scaling mechanism starts with the end user sending a request to the application through a device connected with the Internet. The application forwards this request to the virtual machines which are connected to the load balancer. The applications have to use the auto-scaling mechanism to make the decision of the requirement of virtual machines that is needed based on the request from the user is served (Aslanpour, Ghobaei-Arani & Nadjaran Toosi 2017).

With the high fluctuations in the workload of Internet applications, the main challenge is determining or predicting the future workload of the cloud services (Ghobaei-Arani, Jabbehdari & Pourmina 2016). According to (Calcavecchia et al. 2012), the auto scaling would be required only during the peak hours and for the large-scale applications. However, a centralized approach that is suitable for all kinds of the application must be proposed. Most of the resources are allocated based on the agreed contract between the cloud providers and users, and might end up with users paying more during the scaling (Calcavecchia et al. 2012). Above all, providing a competent server for deployment from the single cloud is the biggest challenge in the cloud environment.

There are many approaches proposed in relation to auto scaling based on CPU load or artificial neural networks, linear regression, ARIMA models (Qiu, Zhang & Guo 2016). The popular service providers, such as Amazon, implement auto scaling using user metrics even it is difficult to tune and optimize (Bitsakos, Konstantinou & Koziris 2018). The automatic provisioning of resource starts with user sending a request to the application to the application server (Arlitt & Williamson 1997).

The application then forwards the request to the virtual machines, which then deploy their auto-scaling mechanism. Based on the load request received, the number of VMs to be used is identified and those VMs start running to accommodate the end user's request (Hassan, Chen & Liu 2018). This ensures that there is no wastage of resources while serving the customer.

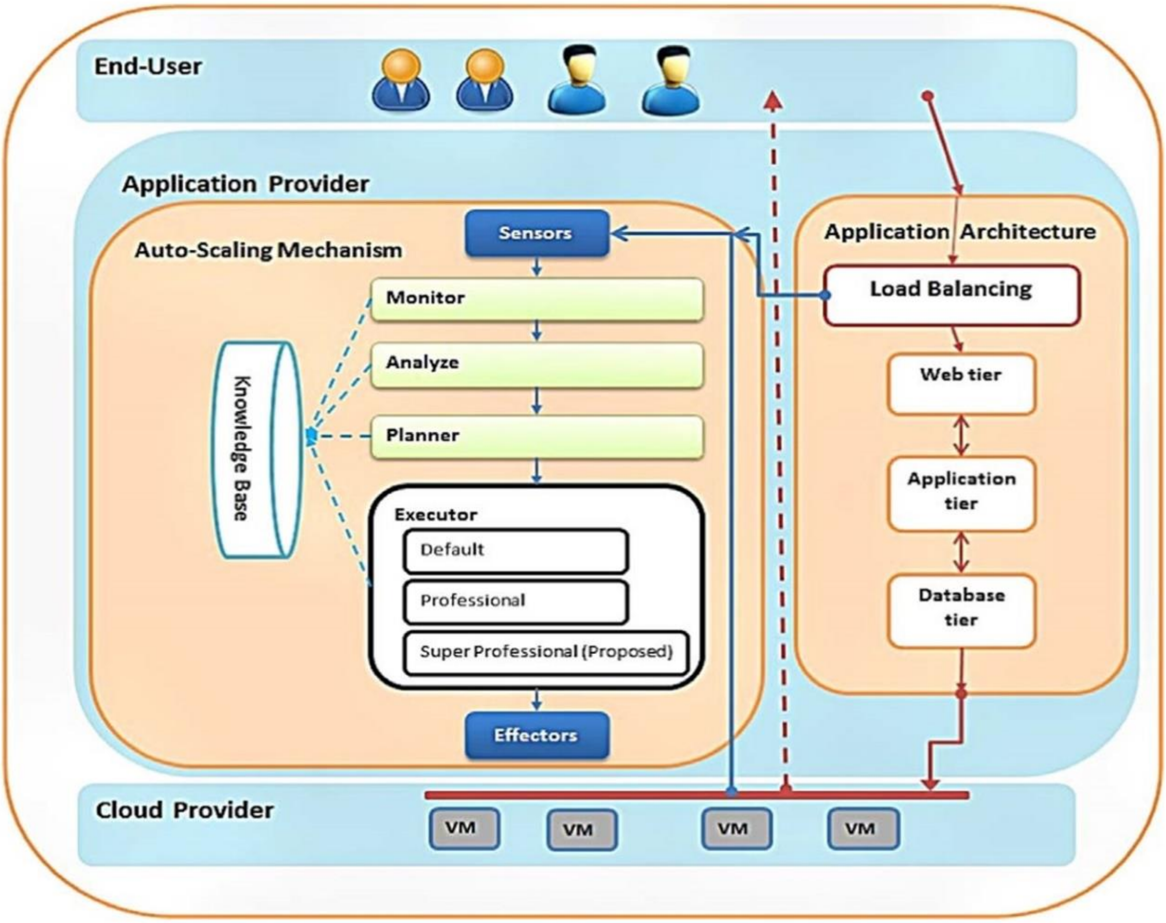


Figure 1:Architecture of cloud-based web applications (Aslanpour, Ghobaei-Arani & Nadjaran Toosi, 2017)

Cloud computing is a motivated technology that provides a distributed infrastructure, which includes environments to host vendor applications, network resources, and build and deploy applications. Many businesses prefer to use these cloud services, which are supplied as VMs by the cloud service providers (CSP). This service is provided by large external data centres and accessed by the business through the Internet (Durkee 2010). This paradigm removes the expense of setting up the servers to host application and other related infrastructure. It not only works on infrastructure also reduces ongoing costs by allowing users to pay only for the resources they use without affecting the performance (Mao & Humphrey 2011). However, these data centers face the worrying challenge of uncontrollable usage of electricity. To protect our environment, it is our responsibility to ensure the energy consumption is reduced. It is projected that by 2020, these data centers will be utilizing 140 billion kilowatts-hour of electricity which will cost around 13 billion US dollars annually (Cheng, Li & Nazarian 2018). To ensure the services are uninterrupted, the data centers are forced to run their servers continuously, irrespective of whether they are needed or not. To deal with providing continuous resources and this energy inefficiency, auto scaling is implemented by service providers. It is an automatic approach that predicts the future requirements and allocates the required resources dynamically and seamlessly when the demand is high, reducing the power consumption (Hassan, Chen & Liu 2018). The cloud providers offer horizontal scaling where more servers are connected parallelly at the same time to do the same work increasing the speed or availability of the logical units and vertical scaling which is the ability to raise the power of the server like increasing the RAM of the computer to improve the speed and performance of the computer (Liu et al. 2014). The automated solution to this horizontal and vertical scaling would benefit both the cloud providers and the users of this cloud services concerning with utilization of resources wisely and cost effectively along with increasing the performance (Liu et al. 2014). Therefore, it is important to build the frequently occurring events in the system so that the cloud systems predict the requirements of the users.

1.1 Significance

To understand workload in advance helps us to proactively control resources, which leads to benefits such as power savings and improved service performance. To efficiently maintain the services to the applications hosted in cloud space, it is required that a mechanism that will release or terminate the

resources based on demand can be built into the system. The tool should perform a series of actions that will monitor the workload and analyse the expected VM. The focus on providing an algorithm to predict the workload to cloud servers will be discussed with the Stakeholder. After overcoming the challenges to determine the workload to the servers, the releasing of the resources can be executed.

The related studies suggest analysing the load to the cloud server is the best solution to decide the VM requirements. When the pressure is measured, the other required criteria to handle the performance is dealt with by itself. While many different models can perform the task of auto scaling, it can still fail on load fluctuations. Load fluctuations can occur when there is more request to the applications at a specific time. The cost of using the resources will increase when more servers are used in order to maintain the performance. Therefore, it is essential to analyse the load to the server for each of the various events so that the most appropriate auto scaling model can be deployed.

1.2 Motivation

To improve the service of provisioning cloud resources and to manage the costs for both the cloud provider and user, it is always a better method to release the cloud resources as and when they are identified as being needed, either through horizontal or vertical scaling, rather than have them available at all times. With the many models that exist in the field of cloud computing, identifying the load before releasing the resources has been successful. Moreover, it is the auto scaling that ensures the required sources of services are supplied seamlessly based on increases or decreases in demand. The automated solution to this horizontal and vertical scaling would benefit both the cloud providers and the users of this cloud services concerning with utilization of resource wisely and cost effectively along with increasing the performance.

In this thesis, I will mainly discuss how the cloud service providers can deal with the varying loads and provide the disrupted services cost efficiently. With such diverse aspects in cloud services that could be considered, the focus of this research will be to predict workload with the framework developed. While focusing on automatically scaling the resources, the quality of the service is ensured to be satisfying and adheres with the SLA between the cloud service providers and users. The main idea of auto scaling is to reduce the human efforts required to configure the system, which in turn reduces the cost involved when using the cloud facilities.

1.3 Research questions

In order to achieve the consistent supply of the cloud service, this thesis will answer the following research questions.

RQ1. How to identify the load to the cloud servers and determine the required resources to supply while ensuring that services are provided disruptively? Identifying workload to the server is always a challenge due to load fluctuations as the load is not easily predictable. The load increases during peak times and the drastically decrease during non-peak hours. Hence, identify load to cloud using an intelligent technique is required to benefit both the cloud providers and users.

RQ2. Which framework to propose to implement the auto scaling? While workload prediction is crucial in providing cloud services, it also important to develop a framework that can calculate the workload.

RQ3. Why the workload prediction model works better than any other model? There are many successful models in existence but predicting the future load by identifying the resources required in order to release the cloud services will always be of greater benefit.

1.4 Objective and Aim

The objective of the thesis was to identify a method that will perform releasing the cloud resources effectively after identifying the workload to the cloud servers. The cloud resources that comfortably address and delivery the needs of any business are called Virtual Machines (VM). The Virtual machines include a wide range of computing solutions that are popularly known as "Anything as a service" (XAAS). Therefore, any computing solutions required can be delivered by cloud services that enables the business applications to run according to the end users' demands. Auto scaling is the feature within cloud services that ideally provides these resources as and when they are required. The increased usage of Internet users is forcing the cloud service providers to maintain a quality service while ensuring the cost is kept low. To effectively maintain the services to the applications hosted in the cloud space, it is required that a mechanism that will release or terminate the resources based on demand can be built into the system. The tool should perform a series of actions that will monitor the load and analyse the expected VM. The focus on providing an algorithm to predict the workload to cloud servers will be discussed with the Stakeholder. After overcoming the challenges to determine the load to the servers,

the releasing of the resources can be executed. The main objective of the framework is to develop a model that will release the cloud resources based on demand automatically. The aims of the research is as follows:

RQ1. To develop a framework that predicts the load to the servers and then calculates the amount of resource to release. Using this framework, the efficiency of the business applications will be improved along with reducing the cost to the customers. To provide these VM's based on demand, it is essential to identify a method that can predict the workload to the cloud servers. While predicting the load to servers, the required unit of activities and the corresponding utilization of resources can also be predicted. Above all, the performance of the system can also be handled appropriately. The related studies suggest that analysing the load to the cloud server is the best solution when deciding the VM requirements. When the pressure is measured the other required criteria to handle the performance is dealt with by itself. While many different models can perform the task of auto scaling, it can fail when there are load fluctuations. The cost of using the resources will increase when more servers are used always in order to maintain the performance. So, it is essential to analyse the load to the server on various events would work better than any other models.

RQ2. To prove using machine learning that prediction models work better than describing models. Five describing models have been investigated and compared against the prediction model. After predicting the workload, it is required to test and identify the future load to the server. The proposed approach in this thesis is capable of dynamically adapting to any unexpected events and fluctuation in workload while calculating the workload.

RQ3. The third aim is to prove that the future workload to release the virtual machines can be identified by workload prediction than any other methods.

1.5 Organisation of thesis

This thesis is presented as follows. Chapter 2 presents background concepts and literature review related to the topic of cloud scaling, auto scaling, time series prediction, linear regression, spot instances, edge computing and the deep learning mechanism. Chapter 3 presents the proposed framework for auto scaling. Chapter 4 introduces time series methods such as ARIMA, Exponential smoothing. Chapter 5 presents machine learning methods such as Linear regression, Naïve and forward propagation. Chapter 6 presents the experiments and results, using two data sources along with the data source details. Chapter 7 concludes the thesis and outlines the future research direction.

Chapter 2

2. Literature Review

2.1 Introduction

This chapter provides the necessary background information for the thesis. Section 2.1 presents a description of workload prediction, the challenges of predicting workload and the importance of auto scaling. (Have to fit according to the subtopic, refer the sample).

Researchers have identified that the scientific applications need a huge number of loosely coupled tasks to be worked out effectively. These applications are classified as two different types: bag-of tasks and workflow tasks (Choi et al. 2015). They have identified that the available computing framework is becoming more complicated with various types like heterogeneous, clusters and private clouds. Therefore, they have considered an auto scaling method that mainly focuses on the CPU utilization and have proposed an extended version of auto scaling that suits heterogeneous frameworks (Choi et al. 2015). There are variety of instance types provided by the cloud providers depending on the demand based on size and prices (Sedaghat, Hernandez-Rodriguez & Elmroth 2013).

With the growing use of technology (Yan, Yu & Long 2019) and data, using a linear model to implement auto scaling would not be enough to meet the demands. With number of factors to consider in determining the resource required, there is a need for a model that can predict the workload based on training the network. Deep learning is a part of machine learning which ensures the calculations are made having considered the various factors that could affect the business applications (Brownlee 2019). The aim of this paper is to use a deep learning algorithm to predict the workload with which a framework could be developed that could handle the provisioning of cloud resources dynamically. This framework would process the user request efficiently and allocate the required virtual machines. As a result, an efficient dynamic method of provisioning of cloud services could be implemented, supporting both the cloud providers and users. Table 1 is a sample of provisioning of resources from Azure. https://azure.microsoft.com/en-us/pricing/details/virtual-machines/linux/

	3 Years Reserved	1 Year Reserved	Pay as you go
Burstable VMs – B1S	$0.009/hour	$0.013/hour	$0.015/hour
Compute optimized – Fv2	$0.063/hour	$0.10/hour	$0.17/hour
General Purpose – Dv3	$0.074/hour	$0.11/hour	$0.192/hour
Memory optimized – Ev3	$0.097/hour	$0.151/hour	$0.266/hour

Table 1: Sample of provisioning resources in Azure

2.2 Auto Scaling

Auto scaling is defined as the quantity of computational resources served based on the load to the server where the user pays for the amount of servers used (Wikipedia 2019). Unlike the traditional method of having a physical server, cloud servers are reliable, secured and maintainable. Both small businesses and large businesses now rely on cloud computing for their data related activities. When the data is transmitted through the cloud servers, the need for the release of the data also increases. Moreover, the data is supplied on the virtual machines supplied by the cloud servers. To efficiently use the resources of cloud computing, auto scaling features have to be implemented (Calcavecchia et al. 2012), so that full advantage of the cloud resources the auto scaling features are used. The major benefit of auto scaling is being able to pay according to the use of the resource and the demand of the request.

There are diverse types of Virtual Machines available in the market to host the business applications. Table 1 provides a sample of pricing and what it could cost the end users to use the service provided. The table also indicates the costing depends on the duration the service is used. The challenge is selecting the most suitable service to handle the request coming to the server. With the wide range of choices available, it is very important to select one that not only best suits the budget but also has the capability to handle the workloads.

2.3 Types of Scaling

While auto scaling is a feature provided by cloud services to add or remove the resource based on necessity, it can be implemented either by changing the partition of resources, such as the CPU, memory and storage inside a VM, or by adjusting the amount of VM instances with certain standard memory, as shown in Fig. 2 (Wang, Chen & Chen 2012). The former is known as vertical scaling and later as horizontal scaling. As shown in Fig. 2, increasing the power of a physical server is vertical resizing and adding additional similar virtual machines is horizontal resizing. The authors (Liu et al. 2014) suggest that under different circumstances either of the methods could be used. At certain times, the cloud computing fails to best serve the end users, as single cloud is not efficient (Liu et al. 2014). While cloud services are provided in various areas, such as Software as Service, Platform as Service and Infrastructure as Service through virtualization technology, a single cloud is limited to provide these services with certain restrictions (Liu et al. 2014). When a single cloud is unable to meet the expectations, then VM's are created either in the same cloud or in a another cloud (Liu et al. 2014). Vertical scaling is when the VMs are created in the same cloud, while horizontal scaling is when the VMs are in different clouds.

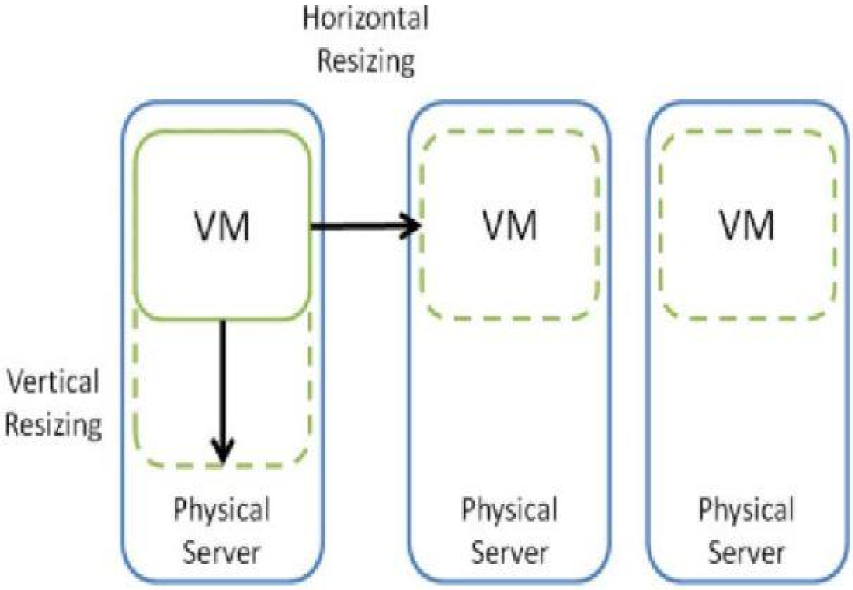


Fig. 2:Types of auto scaling

2.3 The importance and challenges of auto scaling

With increasing development in the Information Technology industry, cloud computing plays a vital role in providing the infrastructure to its users. Most of the IT resources have shifted from physical onsite premises to the cloud computing environments for various reasons such as availability, reliability and elasticity (Kim & Jeong 2016). Compared with the traditional approach of installing an application on their physical servers and then connecting the application through a secured channel, the cloud computing provides easier access to the required resources (Dutreilh et al. 2010). The elastic nature of the cloud computing provides the resources in proportion to the volume request (Mogouie, Mostafa Ghobaei & Shamsi 2015). The aim of auto scaling is to prevent both over and under provisioning of resources (Aslanpour, Ghobaei-Arani & Nadjaran Toosi 2017). A survey conducted by IBM states that around 90% of companies are considering implementing their solution in cloud servers, and there are other surveys that mention that CIO's are evaluating the cloud options for their business (Venters & Whitley 2012). In 2010 *The Economist* magazine estimated Amazon's revenue from between $500 and $700 million, and various other sources expecting the growth to be $241 billion by 2010 (Venters & Whitley 2012). Additionally, cloud users pay only for the service they use for which a high-quality service is provided with security and reliability (Mogouie, Mostafa Ghobaei & Shamsi 2015).

The elasticity feature of cloud computing has lead to the great success in providing various components required as services (Alipour, Liu & Hamou-Lhadj 2014). Another researchers (Ying-chen 2011) detailed the importance of cloud computing in the financial systems and discuss the issues faced by the traditional data centre methods. This paper mentions the wastages of the resources in terms of infrastructure and human resources (Ying-chen 2011).The researchers (Ying-chen 2011) also identified that low utilization of resources is one of the bottlenecks for the financial organisations, where load was not balanced and information could not be shared. Another issue identified, was that the organizations maintained many data centres and were unable to manage these resources as the information was distributed across the various data centres. The writers (Ying-chen 2011) suggested that building a cloud would increase the ratio of usage and decrease the cost, along with creating endless other opportunities to the organizations.

While auto scaling leads to both providing a quality service and reducing the cost to cloud service users, most of the importance is given to the infrastructure level and there is not enough focus on provision at the platform level where the applications are run (Alipour, Liu & Hamou-Lhadj 2014). Additionally, unlike the public and private clouds where difference autoscaling techniques is provided, auto scaling is not supported in the hybrid cloud environments (Alipour, Liu & Hamou-Lhadj 2014).

2.4 Approaches of auto scaling

2.4.1 Linear regression approach

A linear regression model that will predict the work load of the services used in the cloud was also proposed in (Yang et al. 2014) in which the prediction of the workload was based on the request number at time interval. Despite there being a number of existing techniques, the aim of the author was to identify a method which can adjust its model quickly according to the variation trend of workloads (Yang et al. 2014). They observed that the workload trend is linear in a relatively short period of time and used linear regression to solve the problem (Yang et al. 2014).

2.4.2 Spot instances

An approach implemented by Amazon that aimed at selling their unused capacities based on an auction-like mechanism which were called as spot instances was suggested (Qu, Calheiros & Buyya 2016). These spot instances were suitable mainly for fault-tolerance flexible web applications. The cost of this instance was low compared with the cost that is in demand. Moreover, this approach can be used for applications that can be interrupted only (Qu, Calheiros & Buyya 2016). Applications such as background processing and batch jobs can utilise this approach compared to the critical applications. A drawback of this approach is that the spot instances take more time to boot compared to the on-demand instances. The author suggests a heterogeneous approach where a mix of both spot and on-demand instances can be used to meet the end users demand (Qu, Calheiros & Buyya 2016).

2.4.3 Edge computing

Edge computing, which is aimed at providing the ability to process the data within the edge of the network instead of processing from cloud servers, are getting popular currently (Huang et al. 2017). In Internet of things (IoT) related applications, it becomes crucial to distribute the resources. In another paper, deep things, which aims at minimizing memory footprint, especially when getting exposed to parallelism, is proposed (Zhao, Barijough & Gerstlauer 2018). The authors have focused on partitioning and distribution at the early stage using convolutional neural network interference. They propose the File Transfer Protocol (FTP) method firstly and then they develop a distributed work stealing run time system (Zhao, Barijough & Gerstlauer 2018). Since the model works on the edge of the network it becomes essential to predict load and provide an mechanism that can effectively transfer the data as required.

According to (Bob Gill 2019), edge computing is a topology that processes the information close to the edges, and with great promises provided at the early stage there is also a hype attached to this concept. The hype provided to edge computing is shown in Fig. 3 below (Bob Gill 2019). Edge computing is expected to grow, either to in the gaps of the IoT or by extending the IoT.

2.4.4 Time series with regression approach

Researchers (Qiu, Zhang & Guo 2016) suggested that ARIMA is based on statistical analysis and hence designed a prediction model to capture the workload variation patterns. Workload was classified according to the basic resource used in terms of CPU, RAM memory and service request (Qiu, Zhang & Guo 2016). The prediction was based on mean absolute percentage error (MAPE) (Qiu, Zhang & Guo 2016). For a distributed and complex system, it is more important to predict the load based on high level features than to continue working with one particular machine (Qiu, Zhang & Guo 2016). Therefore, a Deep Belief Network (DBN) approach, which has a logistic regression layer to monitor the CPU utilization, was proposed (Qiu, Zhang & Guo 2016). The results showed a prediction accuracy of more than 2.5% and hence was supported for the cloud system in terms of CPU (Qiu, Zhang & Guo 2016).

2.4.5 Deep learning in auto scaling

With increasing usage of cloud computing, there are scenarios where cloud service providers lose their customers for a variety of reasons. The authors Sung etal, identified a data driven iterative churn prediction framework (Sung et al. 2017). They implemented a deep learning approach and proposed the Nascency, Intermediate and Latest (NIL) analysis model. The NIL model showed that identifying customer retention would be profitable and so analysed the data of customers who would stop using their services. They calculated three NIL variables where N is the activities in the initial month of the plan, I is activities in middle month of the plan and L is latest month of the plan. With the challenge where the customer activities could be unpredictably deep, convolutional network was used for investigation. Finally, they clustered the customers who paid and the business team engaged them accordingly (Sung et al. 2017).

In another paper, the deep learning mechanism was used for multimedia analysis and in turn scaling the cloud service (Bao et al. 2018). The paper mainly discussed the technical details of image analysis. This analysis would scale horizontally, based on the user's request. The system was able to provide high accuracy and handled request along with scaling GPU and CPU (Bao et al. 2018). The authors proposed a platform and framework. They assigned a unique key for each image submission.

Once the image is uploaded, the decoding of message received from the customer was authenticated and served. The authors came up with a model which was a combination of Fast RCNN and VGG neural networks (Bao et al. 2018).

Feed forward neural network is used extensively in most of the time series forecast (Frank, Davey & Hunt 2001). Deep learning is proposed mainly for focusing on energy efficiency as well as for providing the cloud resources in data centers (Cheng, Li & Nazarian 2018). According to a deep reinforcement learning (DRL) model, a cloud framework, which aims at reducing the energy used to provision the resources which in turn leads to reducing the cost to the business users, is proposed (Cheng, Li & Nazarian 2018). The model considers two aspects: one where the user request is accepted followed by queuing and the other where energy is minimized (Cheng, Li & Nazarian 2018). DRL cloud consists of user workload, cloud platform, energy consumption and a realistic model all working together to perform while reducing the energy costs for both the cloud providers and end users (Cheng, Li & Nazarian 2018).

2.4.5.1 LTSM and CNN approaches

With more businesses moving towards using the cloud, the energy usage of the data centres is increasing. The data centres are being forced to increase the power of CPU to enable a better service to the end users (Wang et al. 2019). To cope with this challenge, they proposed a model based on the recurrent neural network (RNN) of deep learning and named it Long Short-term memory (LSTM) (Wang et al. 2019). The model works on two phases where in the first phase the system uses the historical time series algorithm to predict the load and then in the second phase, using the results, the values are further enhanced to improve the earlier results making it more precise. The trend on workload was classified to be static, periodic, unpredictable, changing continuously (Wang et al. 2019). Based on the trends of the load the CPU was released. They experimented with the results and identified the request to CPU was not static. The authors introduced their model which works on RNN mechanism where the information is cycled through loops, and decision is made after considering the current and previous inputs (Wang et al. 2019).

Another model based on CNN (Zhao, Barijough & Gerstlauer 2018) was proposed where a cloud-based image analysis platform was implemented. With this model, a unique private key is assigned to each customer for image submission. Once they submit an image, it is uploaded and stored in the cloud server (Bao et al. 2018). When the submission is acknowledged, Redis decides which algorithm is to be called, and which queue it needs to submit the messages to. Models get messages from the corresponding queues, perform analysis, and then pushes the results to Redis. Results may go through certain post-processing, if necessary, and then go back to the customers via RabbitMQ (Bao et al. 2018). Once the user submits an image, two fields are contained in the incoming message: user private key and the images or their URLs. Users are distinguished as per their respective private keys.

Redis also keeps a counter of the number of algorithms in the message. Every time an algorithm finishes the analysis, Redis will deduct one from the counter. When the counter comes to the last algorithm, an output message is pushed to RabbitMQ (Bao et al. 2018). In this paper, we predict the values based on forward propagation.

2.5 Cost benefits in auto scaling

Cloud computing is elastic and scalable, but the two benefits end users expect are low cost and good performance. The budget for implementation quite often happens in the universities and government to implement or invest in the IT sector annually (Mahmud, He & Ren 2015). Hence it is difficult to solve the puzzle of optimisation within the allocated budget. (Mahmud, He & Ren 2015) proposed a BATS, which is budget constrained auto scaling, where in the VM allocation was based on long-term future workload. The results of BATS showed 34% less delay in response compared to even distribution of resources.

2.7 Summary

Many researchers have applied various methods to the applications hosted in the cloud space. However, workload prediction is still an active area of research. And in particular, releasing the cloud resources dynamically, while ensuring the SLA is also achieved, requires a lot more research. Workload prediction is being used extensively in the field of cloud computing in order to maintain low cost for the customers. Many new ideas have been raised to achieve auto scaling, for example spot instances model to provide auto scaling based on an auction-like mechanism and edge computing to provide the resources at the edges of the IoT. However, with time series dataset that mainly focuses on time and request it is better to conduct work with time series algorithm along with using the machine learning techniques. With various approaches inline to predict the workload, time series and machine learning prediction would be more accurate as many criteria is inputted to calculate the future load. As a result, it would be more beneficial to all users along the service providers.

Chapter 3

3. Framework for Auto Scaling

In this chapter, I propose a framework for auto scaling that implements a technique to calculate the workload using time series and machine learning models. Fig. 4 below represents the framework which is based on workload prediction using the deep learning mechanism. The framework consists of a predictor layer, a load-receiver and a load analyser, which takes care of the load entering the system, analyses the load and computes the calculation for all responses received, with the resulting bytes transferred to serve the request.

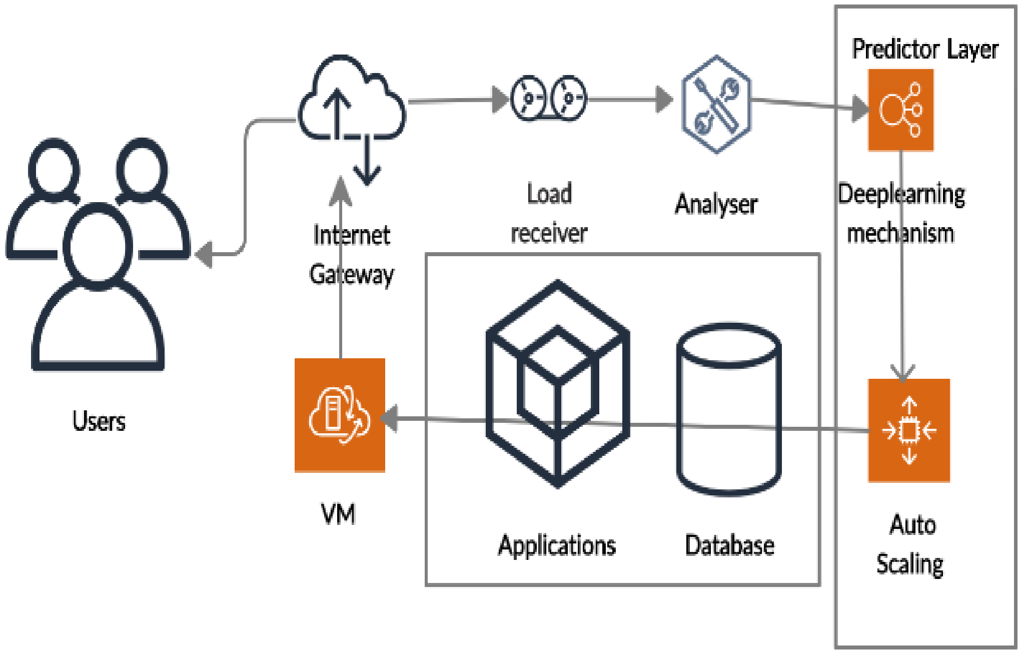


Fig. 3: Proposed framework for load prediction and auto scaling

The load receiver and load analyser layers in the framework receive and analyse the load entering the business application. The load predictor layer accesses the weblog data and implements the forward propagation. The primary aim of the research is to predict the load to the server eventually identifying the future resource requirements. It could be achieved by the various layers discussed below.

3.1 Predictor layer

The predictor layer is loaded with a training data. Using this training data and through forward propagation, the expected load to the server is calculated. To ensure the predication is more accurate, the results computed through forward propagation are taken to compute more precise value using back propagation. After the training data is loaded, the deep learning mechanism control evaluates the workload and sends the details to the load controller, the layer where the final process of provisioning of the VM happens. The load controller controls the VM's and either releases or ceases the resources based on the report it receives from the load predictor. This layer send the data back to the load predictor, which uses the information in the deep learning mechanism layer to work on the most wanted forward and back propagation. This layer sends the related graphs and required details like count of the users and time to release the resources based on the original request, server response and the file are transferred. The load predictor layer has the control over the VM's and either releases or ceases the resources based on the report it receives from the analyser.

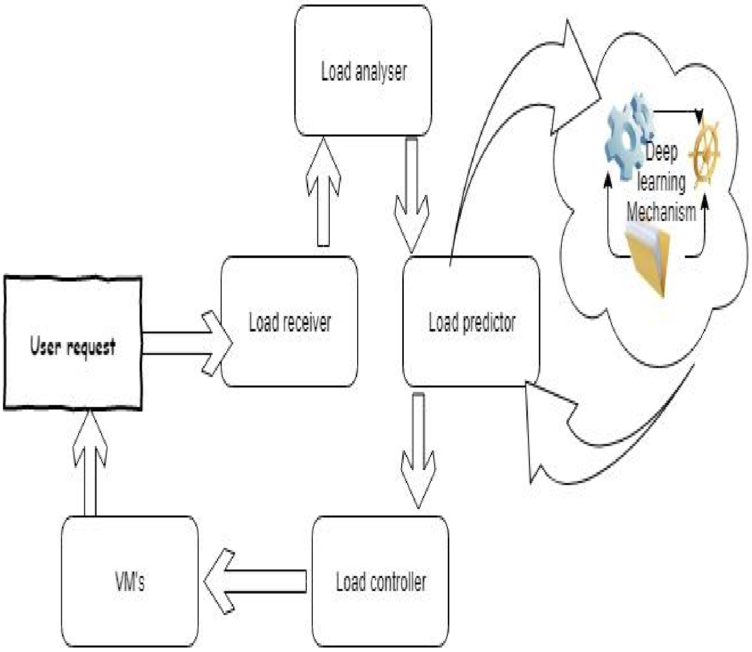


Fig. 4:Representation of proposed framework

3.2 Load receiver

In the above framework (Fig.1 and Fig. 5), the load receiver detects the load entering the cloud server. After receiving the monitored load requests, the data is sent to the load analyser. The cloud providers provision resources through virtual machines, ensuring the quality of service in accordance with the Service Level Agreements (SLA) (Qiu, Zhang & Guo 2016). For example, there might a scenario where there are X number of machines running but only need N number of machines because of low number request received. Now considering this situation if the virtual machines are turned off in order to save the resource and suddenly the unanticipated number of customers uses the same application and this is when the performance of the application becomes down and violates the SLA. The end users access the business application to perform their required task through the Internet. The request for accessing the applications varies at different time intervals and therefore, based on the number of end user requests and depending on the request the VMs should be released. At certain times there might be enormous number of users trying to reach the application layer whereas at other times there might be much fewer or even no requests. Fig.7 shows the workload where users are sending request to NASA servers at a certain time on a day.

3.3 Load analyser

The data from the load receiver is received by the load analyser. The request is analysed and then the load controller handles the release of the resources in the form of Virtual Machines. Based on the forward algorithm in deep learning, the predictor layer calculates the series and predicts the resources that are required on regular basis. With prediction prevailing with the controller, the dynamic provisioning of resources would happen consequently. Without human intervention, the release of resources maintains the quality of service to the end users.

3.4 End user layer

This is the layer where the business applications are hosted and is the layer where the end users requests are sent. This layer may consist of numerous services, applications and databases. To maintain a quality of service, this layer should always be active irrespective of the volume of demands.

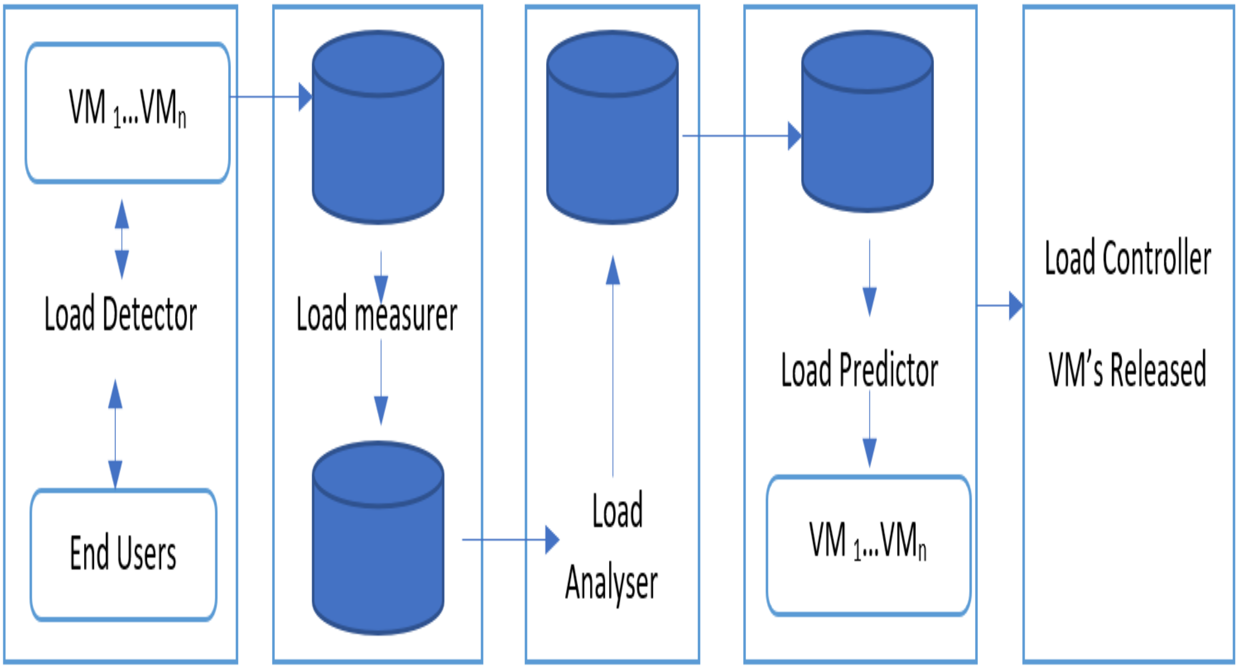


Fig. 5: Flow of data in various layers

3.5 Summary of the framework

Fig. 6 details the flow of data in the various layers. The prediction layer is where the scaling process happens and is the main component of this paper. The predictor layer consists of the life cycle that detects the signal received from the end user, measuring and analysing the request and finally predicting the number of VMs required to handle the demands. This data is sent to the load controller present in the VM layer. The are shared with the load receiver to measure the strength of the load that is received from the receiver.

The load analyser task is to analyse the load that is received. This helps the load controller control the release or ceasing of the VM. The load analyser takes on the responsibility of analysing both resource release and time, and is where the end user requests are analysed and validated. After careful analysis, the data is then supplied to the load controller. The load analyser uses either the auto regressive or linear regression method to analyse the load, so that the analysed data can help the load controller predict the future loads. A linear regression model would be used to measure the load and then proceed with deep learning to work on predicting the future load.

The VM layer is where the virtual machines are ready to auto scale based on the requirements. This layer consists of the N number of virtual machines and the load controller. The load controller receives data from the predictor layer and ensures the required number of VMs are released to meet the end user's demand (Lorido-Botran, Miguel-Alonso & Lozano 2014).

Chapter 4

4. Data Source

4.1 Data source details – NASA

We used the workload characterization study that is composed of the access logs collected from NASA servers. Fig. 2 is the representation of the web users requests to the application server in the month of July. From the workload, it is very clear that the load fluctuates both within the day and from day to day. We have used the web log data which is the record of activity information when a web user submitted the request to a Web Server. The weblog data consisted of IP address, URL, response, bytes transacted as well as date and time stamps.

The NASA workload data consisted of the following information:

- ✓ Host: a remote IP address or domain name – an IP address is a 32-bit host address defined by the Internet protocol,
- ✓ Log name: used to determine a unique Internet address for any host on the Internet which is unused and always an hypon(-),
- ✓ Date and time,

- ✓ Modes of request: GET, POST or HEAD method of CGI (Common Gateway Interface) with HTTP status code returned to the client, e.g., 200 is “ok” and 404 are “not found”,
- ✓ Bytes: The content-length of the document transferred.

The NASA workload represents genuine load changes over time that can be used to compute more authentic and reliable results and conclusions to be used in real-ife environments. The workload comprises 100,960 user requests sent to NASA Web servers. The graph in Fig. 7 shows an extreme fluctuation in this workload. Any business applications in real time would have a similar pattern. To summarise the noted results of usage of server, most load to the web application was less during the early hours, increasing steadily towards the middle part of the day and decreasing towards business closing hours.

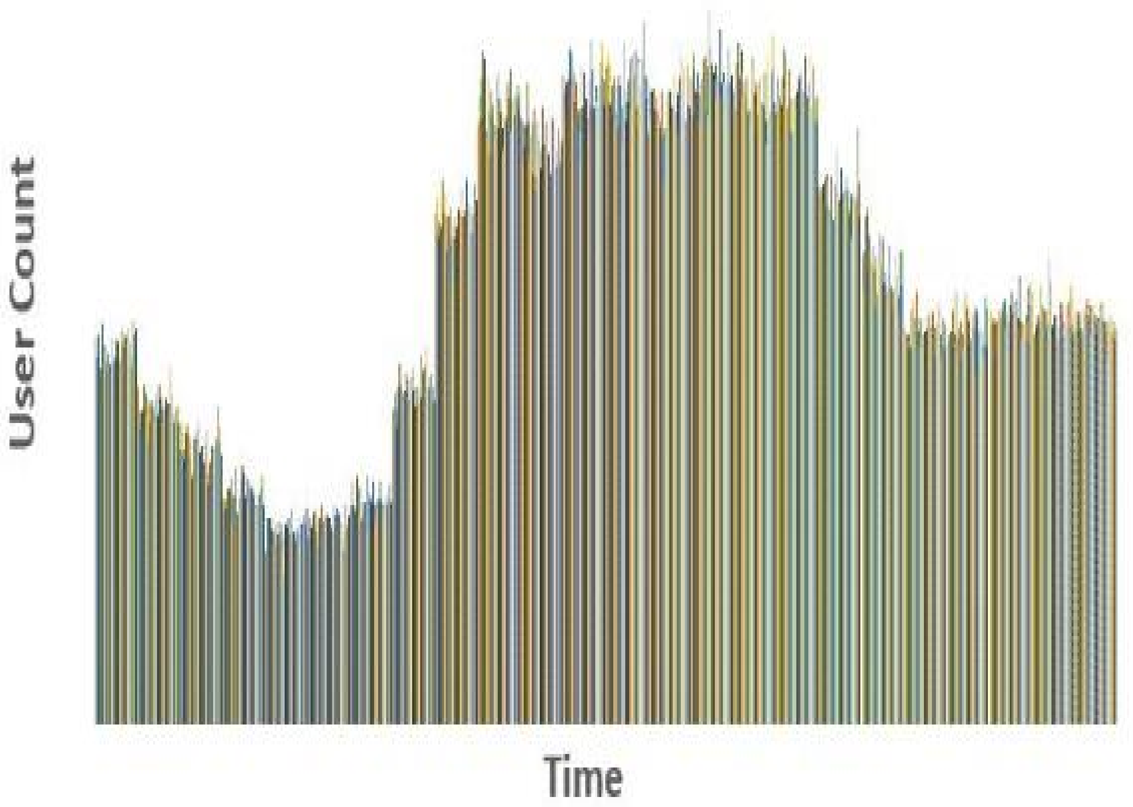


Fig. 6:Workload of NASA dataset (Users Vs Time)

4.2 Data source details – RUET OJ dataset

The second dataset we used to understand the workload characterization study was RUET OJ weblog server data provided in Kaggle and downloaded from Github (https://github.com/shawon100/RUET-OJ/blob/master/LICENSE).

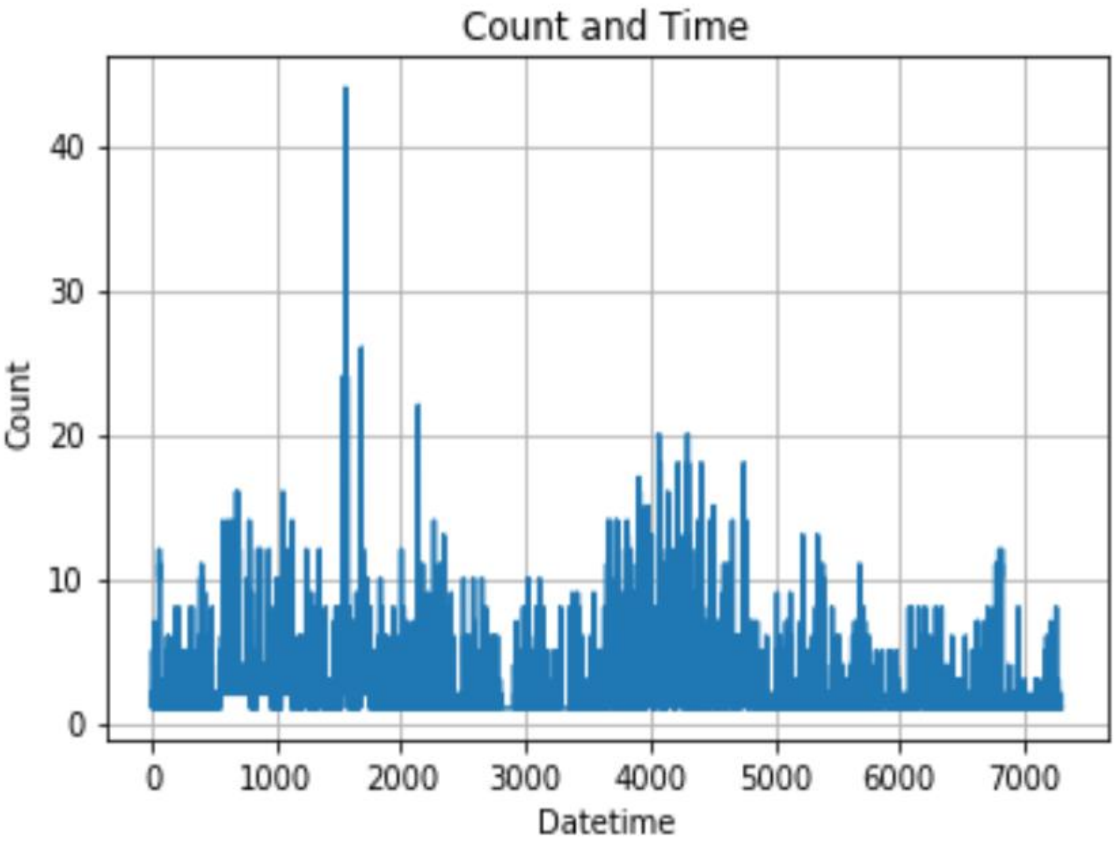


Fig. 7:Workload of RUETOJ dataset (Users Vs Time)

Fig. 8 above shows the number of users at certain datetimes to the server where in the workload is calculated. We can see the fluctuation of load to the server and note that the server is not used very frequently unlike the NASA server (Fig.7). The data that has been taken into account is the activity recorded to the server wherein a request is sent to the server and a response provided by the server. The raw data of this web access log consisted of IP address, data, time, URL and status.

An IP address is the user who sent the request to the server at a certain date and time to a URL of the website. Once the request is received by the server the response provided back is recorded as the status.

For example, the status 200 indicates GET wherein the server is receiving a set of information from the user. On the other hand, a POST operation is recorded with the code 302 where the server is sending the requested information to the user.

RUET OJ is the 1st Open Source Online Judge Platform of Bangladesh. The developer used HTML, CSS, JavaScript for the front end and PHP, MySQL for the backend. This project was developed solely for the learning and understanding of web applications.

4.3 Analysis of datasets

We used NASA server and RUET OJ datasets to experiment and identify the workload. Both the datasets consisted of the user IP address, date, time, request and status of the transaction to the respective servers. While NASA data was very large with a high volume of data, the RUET OJ dataset was limited to certain dates and had comparatively fewer transactions to the server. But both the dataset proved deep learning methods to be effective when comparing the timeseries methods. While examining both the datasets, one common finding was that the load peaked during the middle hours of a business day.

4.3.1 NASA Dataset analysis

We used the workload characterization study that is composed of the access logs collected from NASA servers. Fig. 9 is the representation of web users requests to the application server in the month

of July. As well as showing the fluctuations of workload,

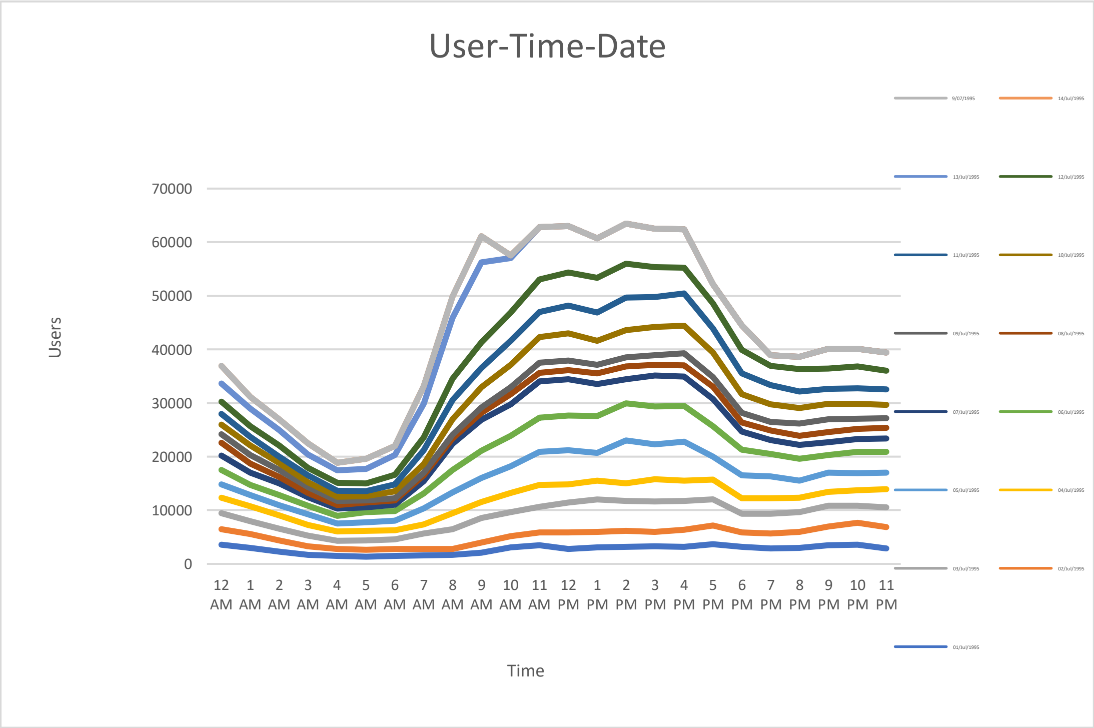


Fig. 8:User Vs Time Vs Date of NASA Dataset

Fig.10 represents time and the quantity of files that was transferred in bytes. The review of data proves the transfer of information is also scattered and is unbalance and nonlinear.

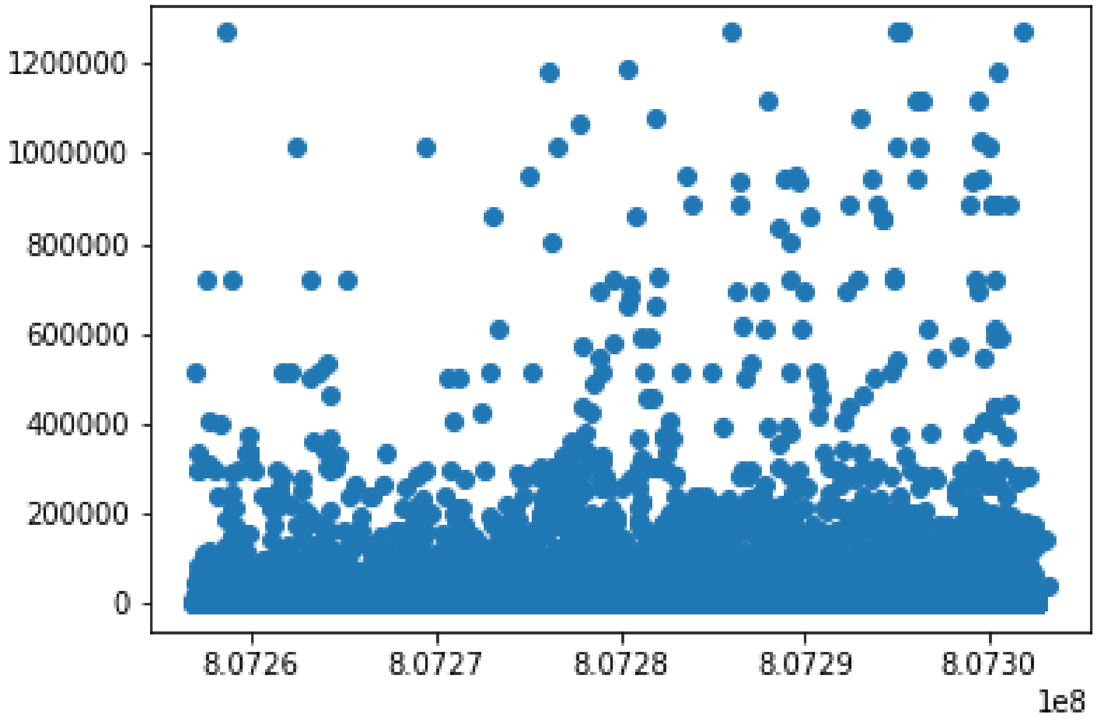


Fig. 9:Time versus files transferred in NASA dataset

the NASA dataset showed the load distribution between users and files transferred as shown in Fig. 11. With the analysis of the NASA dataset, we can understand that there are various loads managed by the server. The loads to the server include user request at certain times of the day and user request of certain files. We can understand from the figures also that at certain times the server works intensively and at other times its workload is much less.

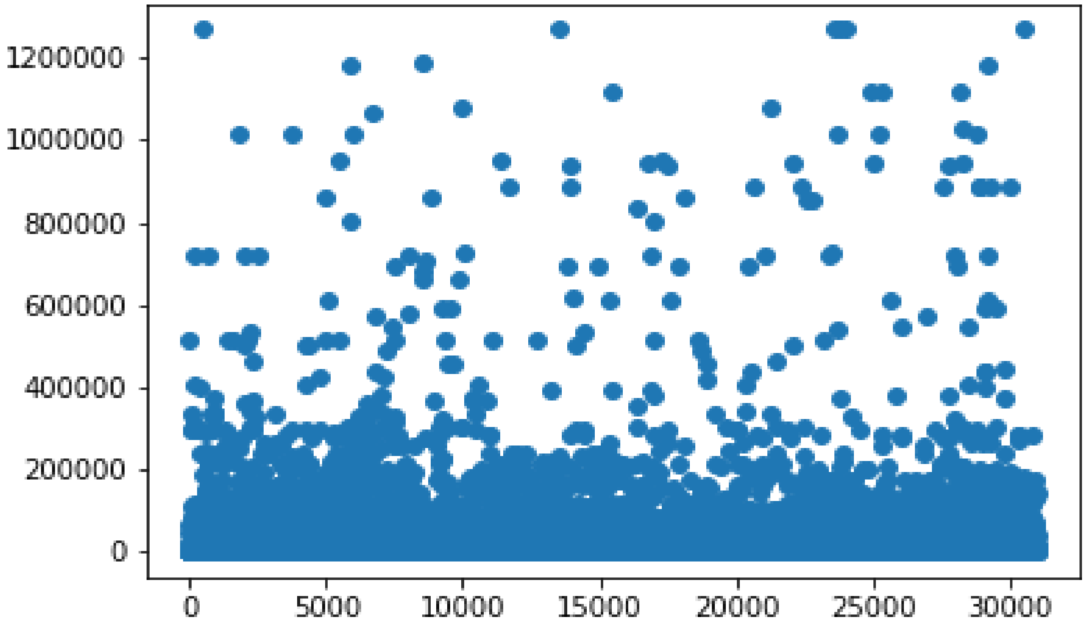


Fig. 10: The load distribution between users and files transferred

From all of the graphs (Fig. 9,10 and 11), we can understand the fluctuation of requests at different times. During these processes of sharing files, sending queries and serving the requests, there might have been some resources that were wasted, which could be avoided by the proposed framework.

4.3.2 RUET OJ Dataset analysis

According to the sources of GITHU, RUET OJ is the first online judge platform of Bangladesh. The dataset provides the users count on a certain date and time to the server. The application is developed by the students of Rajshahi University of Engineering & Technology (RUET). Below is the graph of user count at certain time to the RUET OJ application.

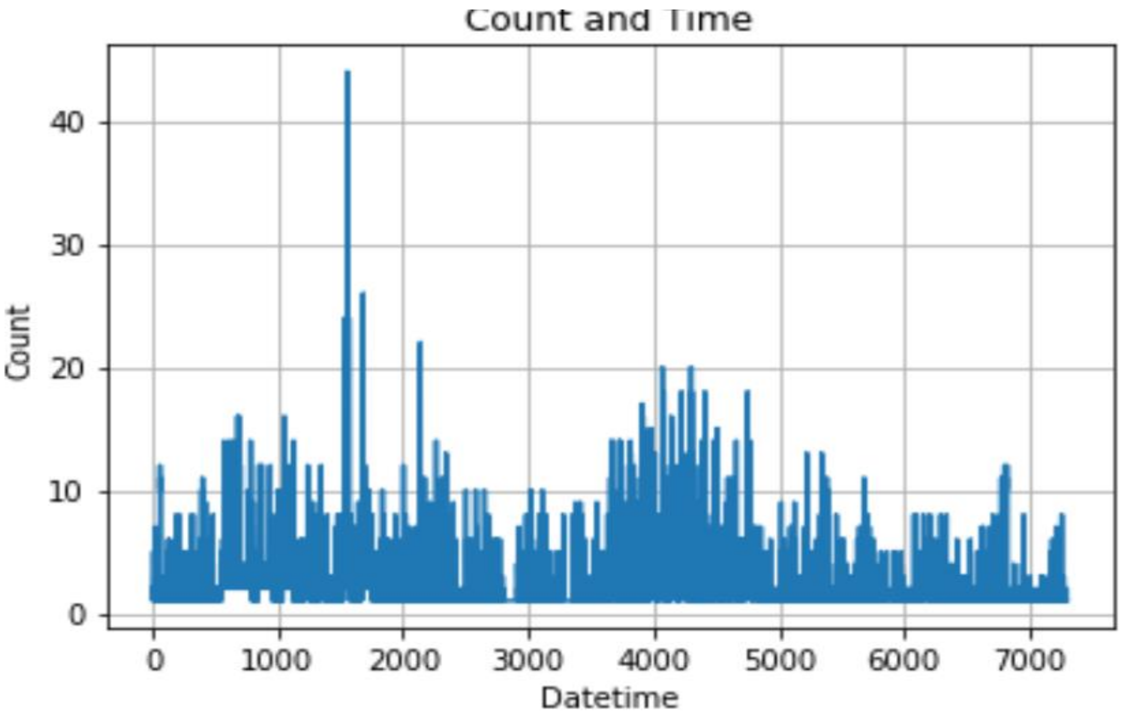


Fig. 11:Number of users at particular date and time

We can understand that the load is at the peak at certain time and low or null at few other times as seen in the above figure (Figure.12). This clearly suggests that a system to monitor the load is required in order not to waste the energy.

Chapter 5

5. Time Series Prediction

Time series is a sequence of linear vectors which is x(t) and t=0,1…n. Here x is the value that changes with certain values of t which is time (Frank, Davey & Hunt 2001). It basically predicts the future values based on current and past values (Sapankevych & Sankar 2009). Time series is used in many sectors, including finance, engineering and economics, to represent the change of a measurement over time. A time series is a sequence of data points, measured typically at successive time instants spaced at uniform time intervals. An example of time series in this thesis context is the number of requests that is send to an application in one-minute intervals. The time-series analysis could be used to find repeating patterns in the input workload or to try to forecast future values.

5.1 ARIMA prediction

Another common and popular time series model is ARIMA, which is Autoregressive Integrated Moving average. While exponential smoothing models are based on a description of trend and seasonality in the data, ARIMA models aim to describe the correlations in the data with each other. ARIMA is used for analysing and forecasting the time series data.

ARIMA is a stochastic modelling approach that can be used to calculate the possible future value lying between two specified limits. To build an ARIMA model, we needed to understand the time series. Followed by that we must identify p, d, q where p means the number of preceding ("lagged") Y values that have to be added/subtracted to Y in the model, so as to make better predictions based on local periods of growth/decline in our data. This captures the "autoregressive" nature of ARIMA.

d represents the number of times that the data must be “differenced” to produce a stationary signal (i.e., a signal that has a constant mean over time) (Abugaber). This captures the “integrated” nature of ARIMA. If d=0, this means that the data does not tend to go up/down in the long term (i.e., the model is already “stationary”). In this case, we are performing just ARMA, not AR-I-MA. If p is 1, then it means that the data is going up/down linearly(Abugaber). If p is 2, then it means that the data is going up/down exponentially. q represents the number of preceding/lagged values for the error term that are added/subtracted to Y, which is the “moving average” part of ARIMA(Abugaber). ARIMA models are typically expressed like “ARIMA(p,d,q)”, with the three terms p, d, and q defined as follows. In terms of y, the general forecasting equation is where ŷt is predicted load:

$$Y_t = c + \phi_1 y_{d\,t-1} + \phi_p y_{d\,t-p} + \ldots + \theta_1 e_{t-1} + \theta_q e_{t-q} + e_t$$

To identify the appropriate ARIMA model for Y, we first began by determining the order of differencing (d) and removed the gross features of seasonality (Abugaber).

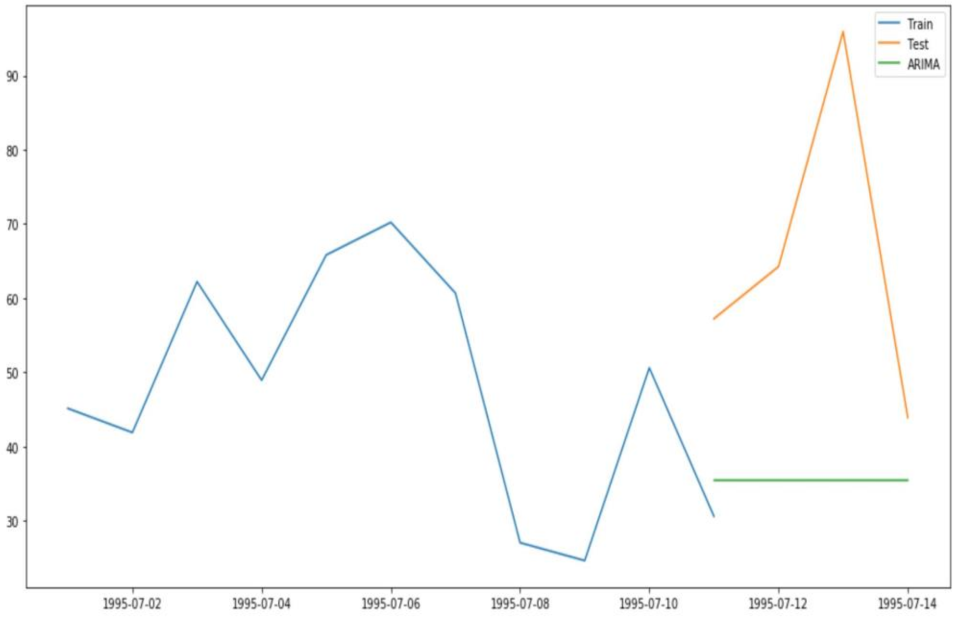


Fig. 12:Workload forecast using ARIMA of NASA dataset

Fig.15 shows the prediction of workload based on the ARIMA method. The ARIMA forecasting for a time series is nothing but similar to linear regression equation. The predictions mainly depend on the parameters (p,d,q) of the ARIMA model. The number of AR (Auto-Regressive) terms (p) and AR terms are just lagging of dependent variable. For example, if p is 10, the predictors for x(t) will be x(t-1)….x(t-10). The number of MA (Moving Average) terms (q) where MA terms are lagged forecast errors in prediction equation. For instance if q is 10, the predictors for x(t) will be e(t-1)….e(t-10) where e(i) is the difference between the moving average at i'th instant and actual value. Finally, the number of Differences (d) are the number of nonseasonal differences, i.e. in this case we took the first order difference. So, either we could pass that variable and put d=0 or pass the original variable and put d=1. Both will generate the same results. The Fig.6 is generated by passing the value d=0. An important point with ARIMA is how to determine the value of 'p' and 'q'.

Another researcher (Yang et al. 2014) compared predicted workloads with actual workloads, and a set of alternative workload prediction algorithms using ARIMA. They compared the load with a second order autoregressive moving average method filter (ARMA) with the equation below (Equation.2)

$$Yt+1 = \beta * Yt + \gamma * Yt-1 + (1 - (\beta + \gamma)) * Yt-2$$

Equation 1:General form of ARMA

The value for the variables β and γ were given by the values 0.8 and 0.15, respectively. The graph in Fig. 14 is the ARIMA model for the RUET OJ Dataset.

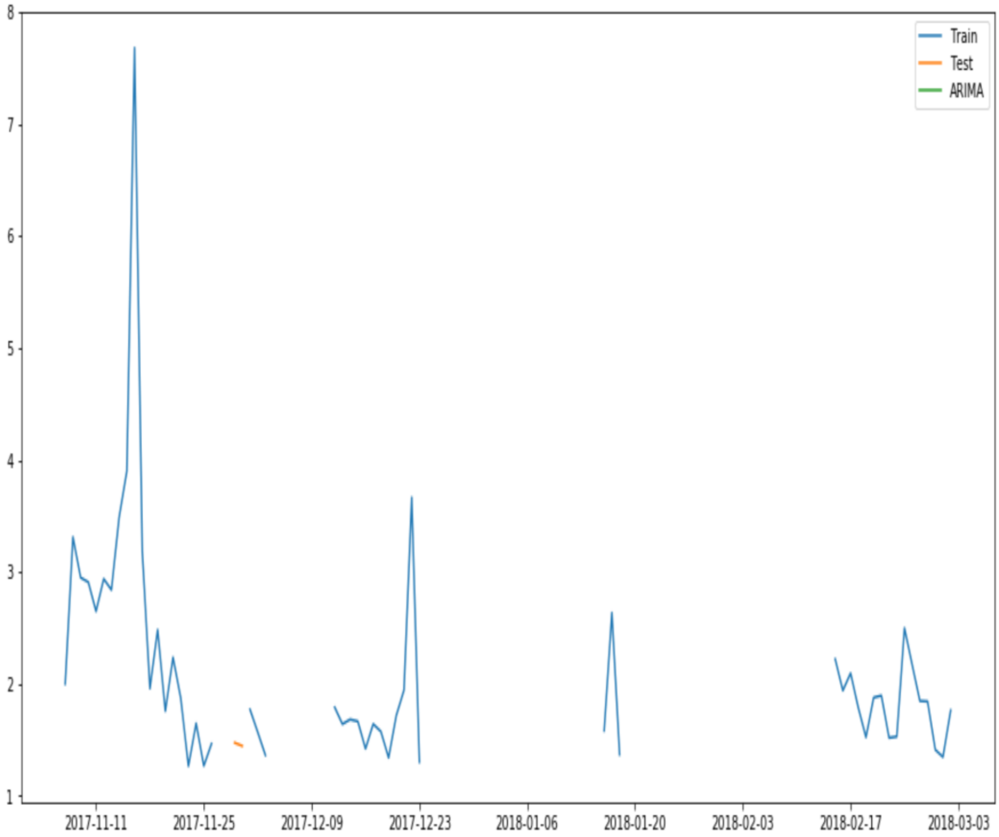


Fig. 13:ARIMA prediction for the RUET OJ dataset

The above graph (Fig.14) represents the results of the ARIMA method used to forecast the future number of users to the server. We have used the RUET OJ dataset and divided the dataset into train and test data. Using this train and test data we have predicted the future load. The future load expected to this server is comparatively less or null to that of the busy NASA server.

5.2 Exponential smoothing

Exponential smoothing methods are forecasting models that calculate the weighted averages of past observations to forecast new values. The main aim of this method is to give more importance to recent values in the series. Thus, as observations get older (in time), the importance of these values gets exponentially smaller. Exponential smoothing methods combine error, trend, and seasonal components in a smoothing calculation. Each term can be combined either additively, multiplicatively, or be left out of the model. These three terms are Error, Trend, and Season (Daitan 2019).

Exponential smoothing is used to make short term forecasts. Longer-term forecasts using this technique can be unreliable, since the older values are not given the same importance as the newest values. The following are the different techniques of exponential smoothing.

Simple (single) exponential smoothing uses a weighted moving average with exponentially decreasing weights, while Holt’s trend-corrected double exponential smoothing is usually more reliable for handling data that shows trends, compared to the single procedure. Triple exponential smoothing (also called Multiplicative Holt-Winters) is usually more reliable for parabolic trends or data that shows trends and seasonality.

In our experiments we used the exponential smoothing approach, which is used for univariate data that can be extended to support data with a systematic trend or seasonal component in time series predictions. It uses the forecasting methods that are similar to prediction that is a weighted sum of past observations, but the model explicitly uses an exponentially decreasing weight for past observations. It uses the following formula (Equation.1):

$$a_t = \alpha(X_t - F_{t-s}) + (1 - \alpha)(a_{t-1} + b_{t-1})$$

$$b_t = \beta(a_t - a_{t-1}) + (1 - \beta)b_{t-1}$$

$$F_t = (X_t - a_t) + (-)F_{t-s}$$

Equation 2:Exponential smoothing general form

Here α, β, and γ are smoothing constants that are between zero and one. Again, (alpha)α gives the y-intercept (or level) at time t, while bt is the slope at time t and S is the given value of period. As we can see, 1−α is multiplied by the previous expected value which makes the expression recursive. When the Search Method is set to Specified Value, this option specifies the value of alpha. Alpha is the smoothing constant for the level of the series.

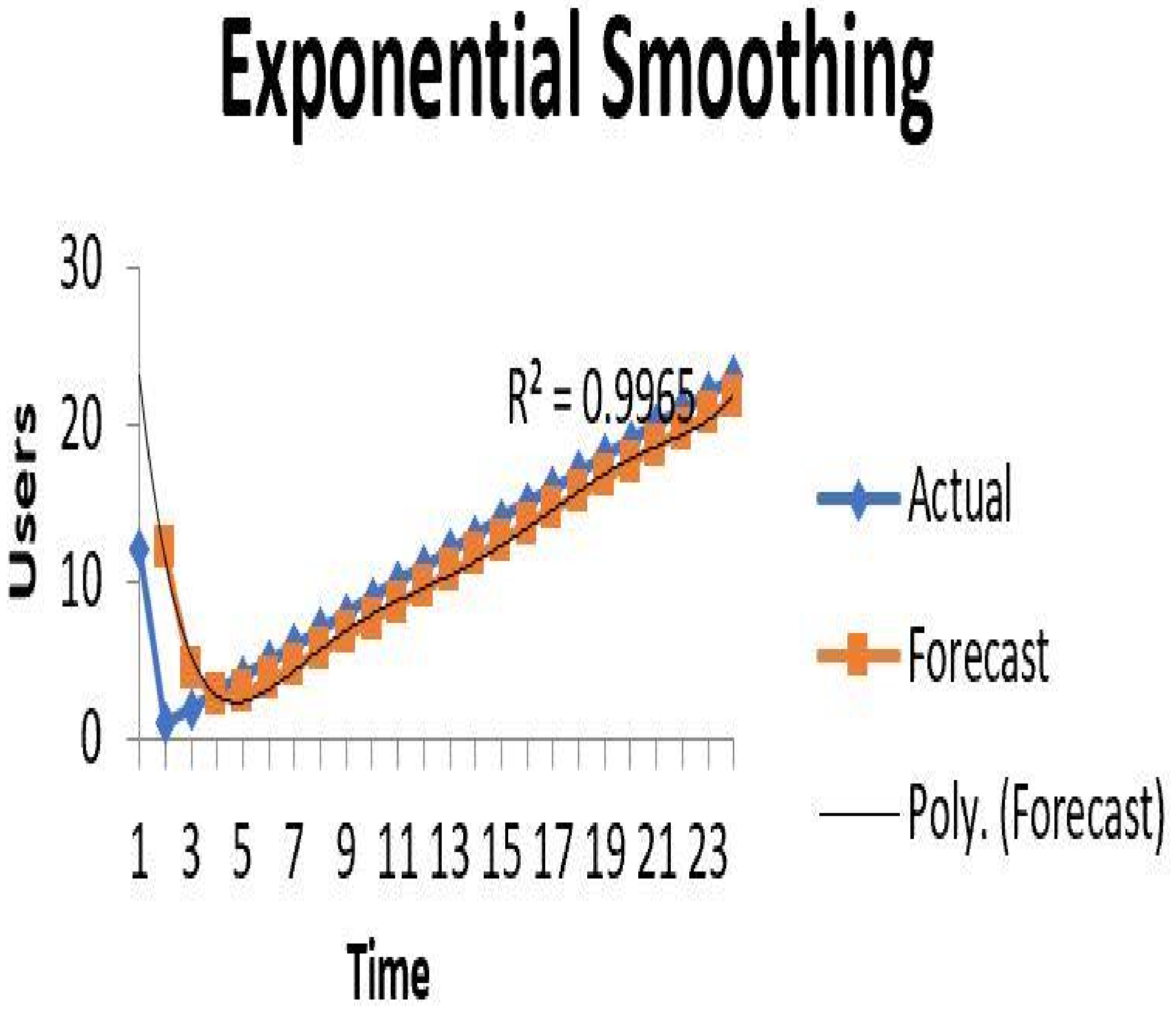


Fig. 14:Workload prediction using exponential smoothing of NASA dataset

The limits of this value are zero and one. Usually, a value between 0.1 and 0.3 are used. As the value gets closer to one, more and more weight is given to recent observations. We have used the smoothing constant value as 0.3. Fig. 16 is the prediction of workload based on an exponential smoothing approach.

Chapter 6

6. Machine Learning Prediction

6.1 Linear regression

Linear regression is used to help predict future values from past values and is commonly used as a quantitative way to determine the trend of the users and when the server is overloaded. It uses the least squares method to plot a straight line through count of users, so as to minimize the use of server when the load to the server is less. This linear regression indicator plots the trendline value for each data point. We used dependent and non-dependent variables. The dependent variables are the ones that need to be predicted and the non-dependent variables are the factors influencing the dependent variables. The linear regression model calculates the current values of series against the prior values in the series. The general form of linear regression is given as follows (Equation.1):

$$Y_{t+1} = \beta_1 + \beta_2 * X_1$$

Equation 3:General form of linear regression

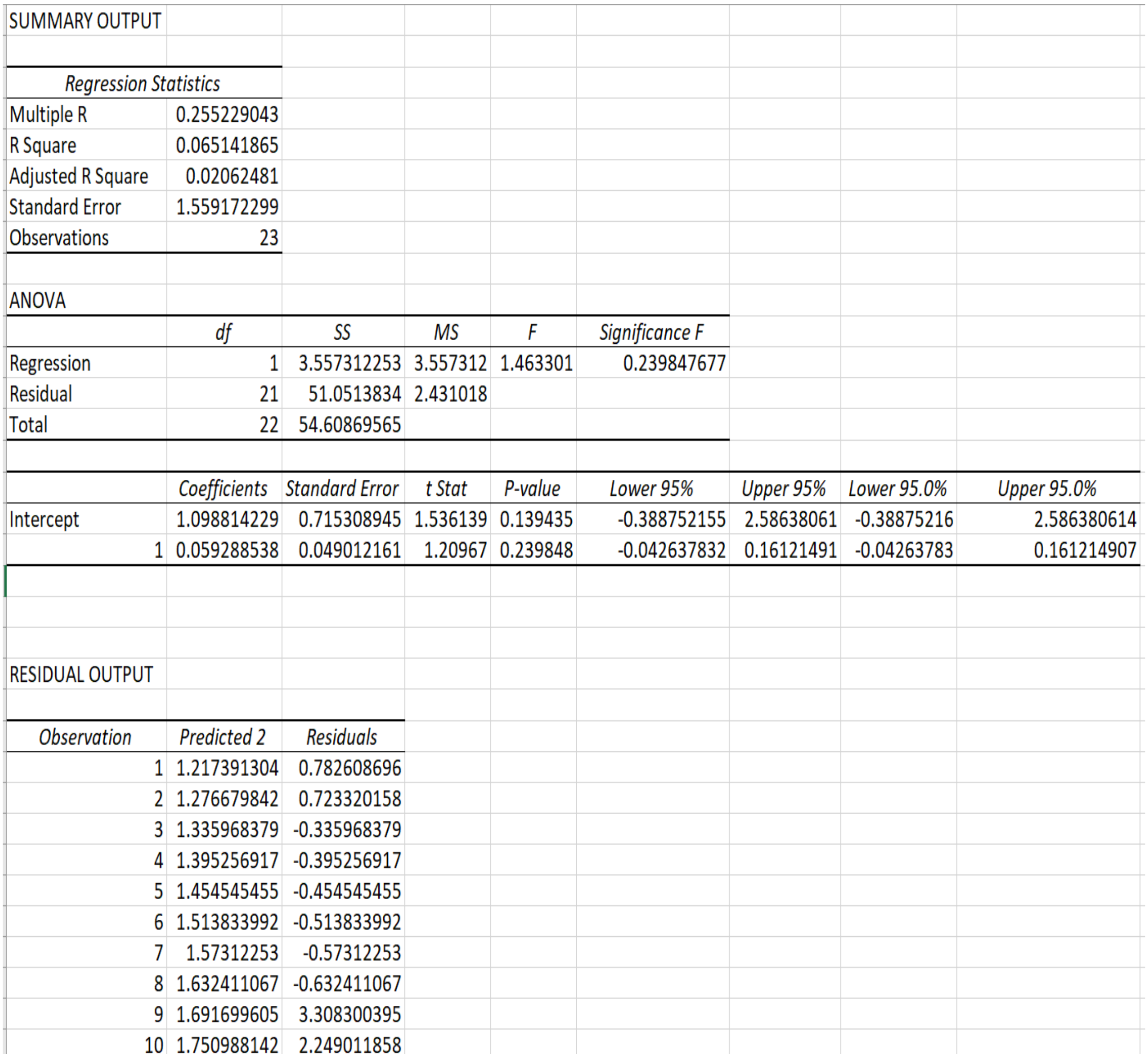
where t is indexes, Y1 is the incoming workload and X is the actual value of the instance at that moment.

The load can also be measured using the moving average method or exponential smoothing methods. In moving average, the mean of the n last values is calculated, while the exponential smoothing method decrease values in each value of time series. In our case, the number of users was the dependent variable, and the date and time were the non-dependent variable. Using the regression analysis, we calculated the changes happening to the server as the time changed. We used a simple regression where we had one dependent and one non-dependent variable. Below is the equation for linear equation,

$$y = bx + a + \varepsilon$$

where y is dependent variable, x is the non-dependent variable, a is the Y-intercept, which is the expected mean value of y when all x variables are equal to 0. On a regression graph, it is the point where the line crosses the Y axis. b is the slope of a regression line, which is the rate of change

for y as x changes and ε is the random error term, which is the difference between the actual value of a dependent variable and its predicted value. Below is the sample of the results of linear regression for the RUET OJ dataset.



SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.255229043
R Square	0.065141865
Adjusted R Square	0.02062481
Standard Error	1.559172299
Observations	23

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	1	3.557312253	3.557312	1.463301	0.239847677
Residual	21	51.0513834	2.431018		
Total	22	54.60869565			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	1.098814229	0.715308945	1.536139	0.139435	-0.388752155	2.58638061	-0.38875216	2.586380614
1	0.059288538	0.049012161	1.20967	0.239848	-0.042637832	0.16121491	-0.04263783	0.161214907

RESIDUAL OUTPUT

Observation	*Predicted 2*	*Residuals*
1	1.217391304	0.782608696
2	1.276679842	0.723320158
3	1.335968379	-0.335968379
4	1.395256917	-0.395256917
5	1.454545455	-0.454545455
6	1.513833992	-0.513833992
7	1.57312253	-0.57312253
8	1.632411067	-0.632411067
9	1.691699605	3.308300395
10	1.750988142	2.249011858

Fig. 15:Sample output of Linear regression for RUET OJ dataset

From the above Fig.17 multiple R is the Correlation Coefficient that measures the strength of a linear relationship between two variables. The correlation coefficient can be any value between -1 and 1, and its absolute value indicates the relationship strength. The larger the absolute value, the stronger the relationship. If it is a 1 then it means a strong positive relationship, -1 means a strong negative relationship and 0 means no relationship at all. Adjusted R Square is the R square adjusted for the number of independent variables in the model. Standard Error is another goodness-of-fit measure that shows the precision of your regression analysis – the smaller the number, the more certain you can be about your regression equation. While R2 represents the percentage of the dependent variables variance that is explained by the model, Standard Error is an absolute measure that shows the average distance that the data points fall from the regression line. Observations is simply the number of observations in the model. The next part is ANNOVA which is analysis of variance. Basically, it splits the sum of squares into individual components that give information about the levels of variability within your regression model. The Significance F value gives an idea of how reliable (statistically significant) your results are. If Significance F is less than 0.05 (5%) then we can assume the model is OK. Fig. 17 is the graph for Linear regression using the RUET OJ dataset.

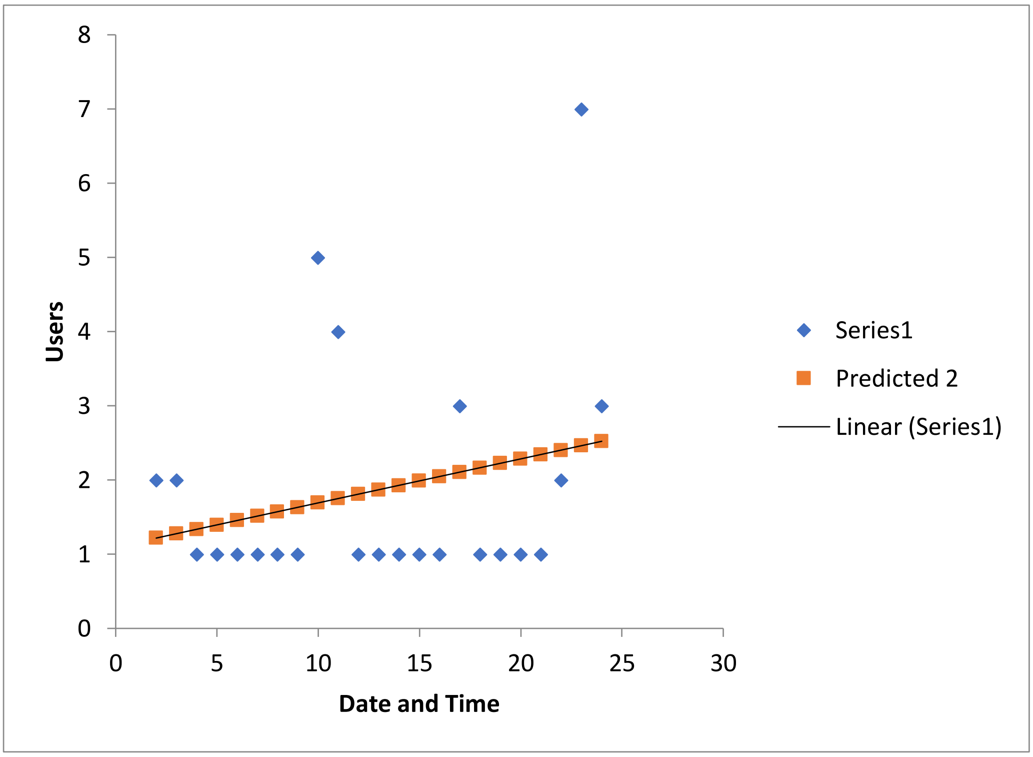


Fig. 16: Linear Regression method for RUET OJ dataset

6.2 Naïve method

Although the naïve Bayes is often significantly more accurate than more sophisticated methods, the chances of estimation it produces can be inaccurate, and it often assigns maximum probability to the correct class. This suggests that its good performance might be restricted to situations where the output is categorical (Frank et al. 2000). It is therefore interesting to see how it performs in domains where the predicted value is numeric, because in this case, predictions are more sensitive to inaccurate probability estimates (Frank et al. 2000). When the training data size is small relative to the number of features, the information on prior probabilities can be used to improve the results.

Firstly, we used naïve approach to predict the workload. It is a very simple forecasting method that uses the most recent observation. It can be implemented in a namesake function, which is the best that can be done for many time series data. Even though it is not a good forecasting method, it can be used as a benchmark for other forecasting methods. To summarise, the naïve method is a forecasting technique which assumes that the next expected point is equal to the last observed point and can be obtained by the formula below:

$$Y\ t = Y\ t\text{-}1$$

where Y t is forecast at time t and Y t-1 is actual data at time t. Fig. 19 is the implementation of the naïve method for the NASA weblog data. This method produced a root mean square error of around 45% and so we can infer that, unlike a stable dataset, the naïve method isn’t suited for datasets with high variability.

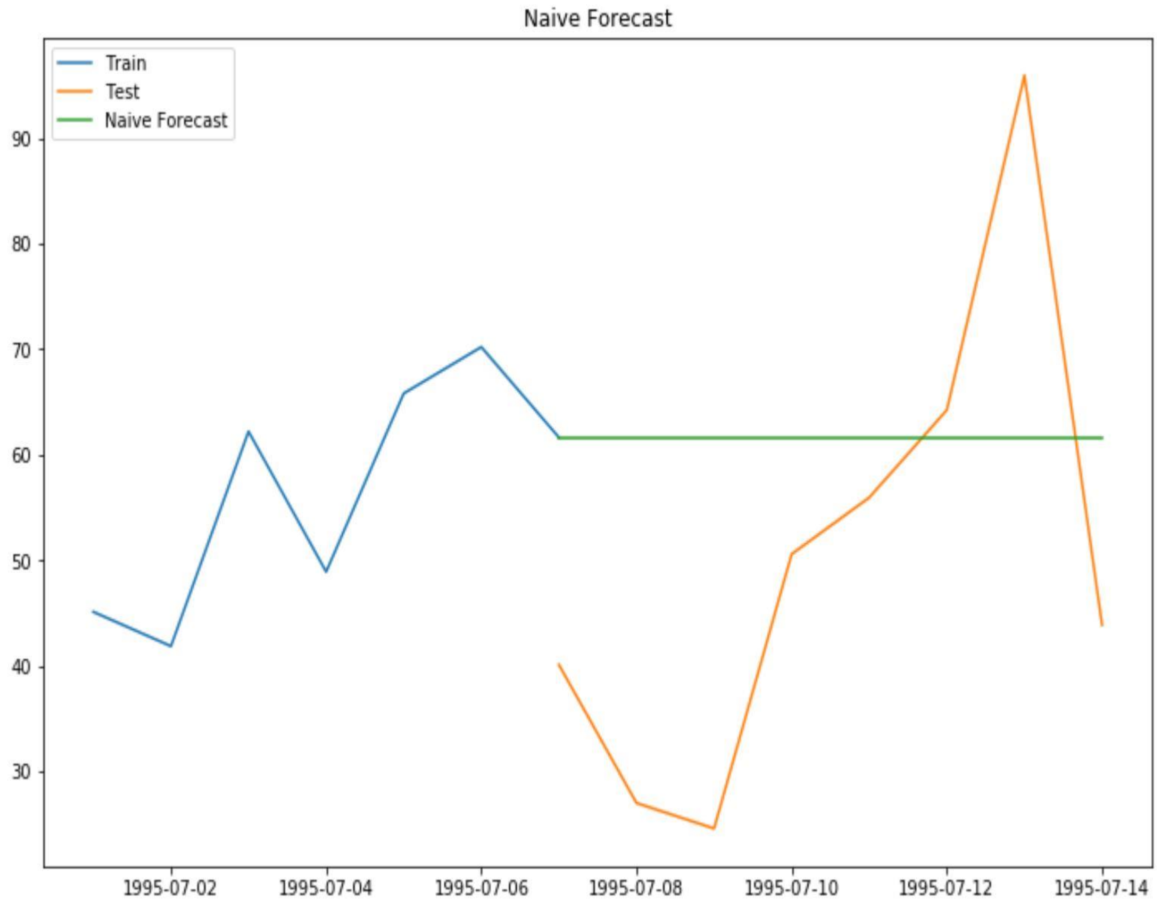


Fig. 17:Forecast using naïve approach for NASA Dataset

We used naïve method to start the forecast so as to get an idea of transactions. Above(Fig:18) is the graph that shows the results of naïve method that identifies the actual load and predicts the load in future to the server. We have divided the data in test and train with a 70%:30% ratio.

The graph in Fig. 18 represents the results of the naïve method, which determined the future value based on the previous period values for the NASA dataset. To understand the results, I calculated the root mean square value as 46%. For naïve forecasts, we simply set all forecasts to be the value of the last observation. Using the naïve method, it is hard to predict the exact values, but it can be used to serve as a benchmark rather than the method of choice. That is, any forecasting methods developed will be compared to these simple methods to ensure that the new method is better than these simple alternatives.

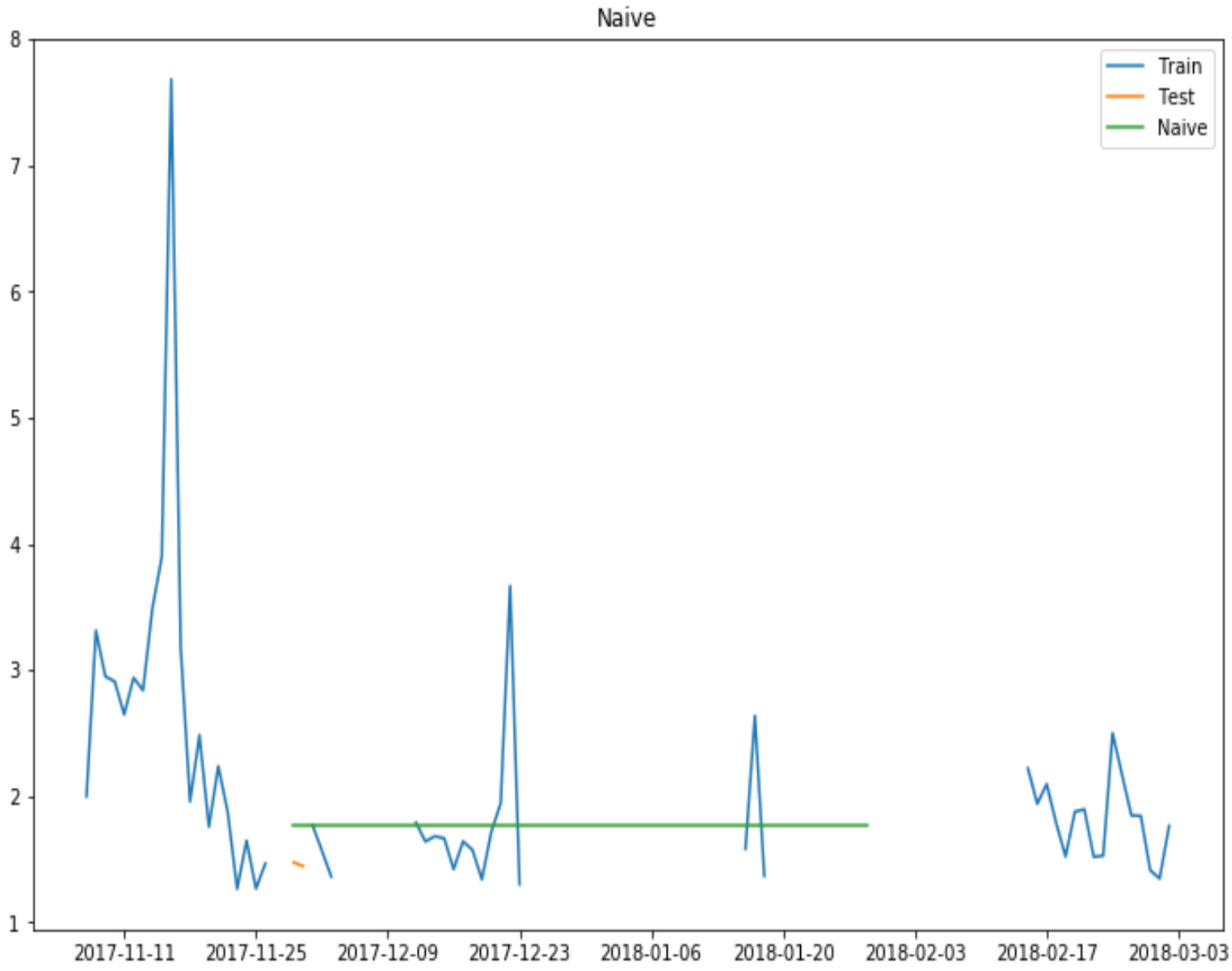


Fig. 18: Naïve forecast for RUET OJ dataset

Naïve forecasts can be used to identify the workload and predict the future load randomly, and the result obtained can be used for getting a clear idea of future load. Fig. 19 is the graph representing the ARIMA method of forecasting for the RUET OJ dataset.

6.3 Deep learning (forward propagation)

Deep learning is a machine learning technique that computes calculation to assist computers to do what they do. A computer model understands to perform various tasks directly from images, text, and both structured and unstructured data. Deep learning models are trained by using a large set of data that contains many layers. Deep in deep learning refers to the number of hidden layers in the neural network. Deep learning networks can have up to 150 hidden layers. With workload variations, it is important to learn the features and relativeness (Qiu, Zhang & Guo 2016), Since deep learning is strongly recommended to learn various patterns, it can also be used to learn the variations in workload data.

Fig. 20 is a basic layout of the neural network. The layout consists of three basic layers such as an input layer, a hidden layer and an output layer. The input layer is the first layer that can take the input values and pass them to the next layer called hidden layer. The output to the hidden layer is provided based on calculation, which is sum of the product of input value and weight. A weight represents the strength of connection between units, which means if the weight of node 1 is greater than node 2 then node 1 has greater influence over node 2 and vice versa. Basically, the weight decides the value of input.

If the weight is zero, then the input has no effect, and if the weight is negative then increasing the input will decrease the output on the hidden layer neurons (nodes), which apply different conversions to the input data. All the nodes in a hidden layer are connected to each node in the next layer, which is the output layer where we can predict the desired number of values in a certain range.

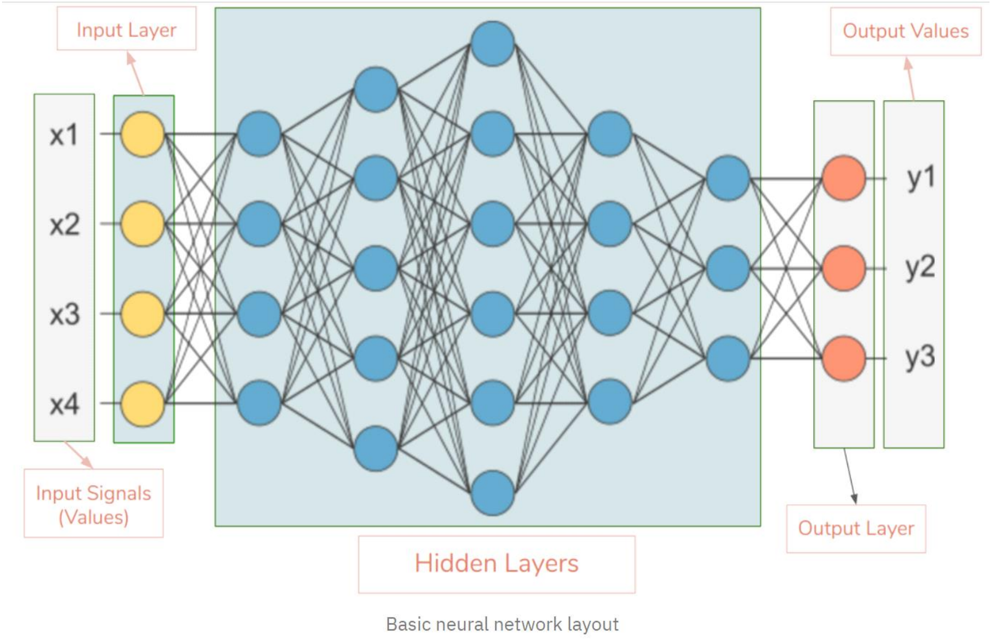
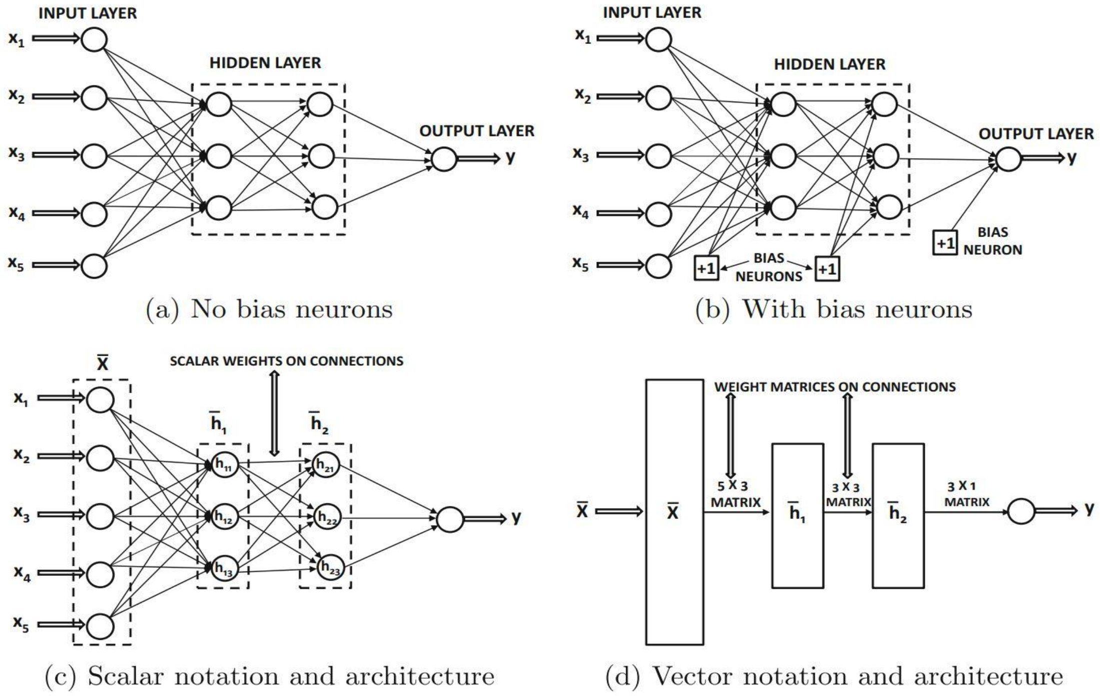


Fig. 19:Basic layout of neural network

In forward propagation, the input values are served to the neural network's first layer without any operation. Then the second layer takes the value and applies multiplication then addition and a performs activation operation. The output of second layer is fed to the next layer which again follows a similar operation to the second layer to produce the results. Next backward propagation is performed with the output value, which is the predicted value through forward propagation. To calculate error, we compared the predicted value with the actual output value. We used a loss function (mentioned below) to calculate the error value. Then we calculated the derivative of the error value with respect to every weight in the neural network. Back propagation uses the chain rule of Differential Calculus. In chain rule, the derivatives of error value are calculated with respect to the weight values of the last layer. These derivatives are called gradients and these gradient values are used to calculate the gradients of the second last layer. We repeated this process until we had gradients for every weight in our neural network. Then we subtracted this gradient value from the weight value to reduce the error value.

The forward propagation can be performed by first initializing the input layer with certain weight, typically between -1 and 1. Then we can iterate through the input-layer to compute the values of hidden layer. This type of architecture in multilayer neural networks is referred as feed-forward networks as they move forward to feed the values for the next layer (Aggarwal 2018). Fig. 21 is the representation of forward propagation.



(a) No bias neurons

(b) With bias neurons

(c) Scalar notation and architecture

(d) Vector notation and architecture

Fig. 20:Basic architecture of feed forward network (Aggarwal, 2018)

With results varying in different methods, we built a deep learning model to predict the workload. We determined the input layer with time and number of requests. To make it simple, we assigned a weight between -1 to 1 for the input layer and calculated the values of the hidden layer and assigned the output in an array. With the values of the hidden layer, we then calculated the predicted values again with assigning similar weights. Fig. 22 represents the workload prediction by training the dataset with forward propagation wherein actual represents the original value and predicted represents the value predicted with the forward propagation

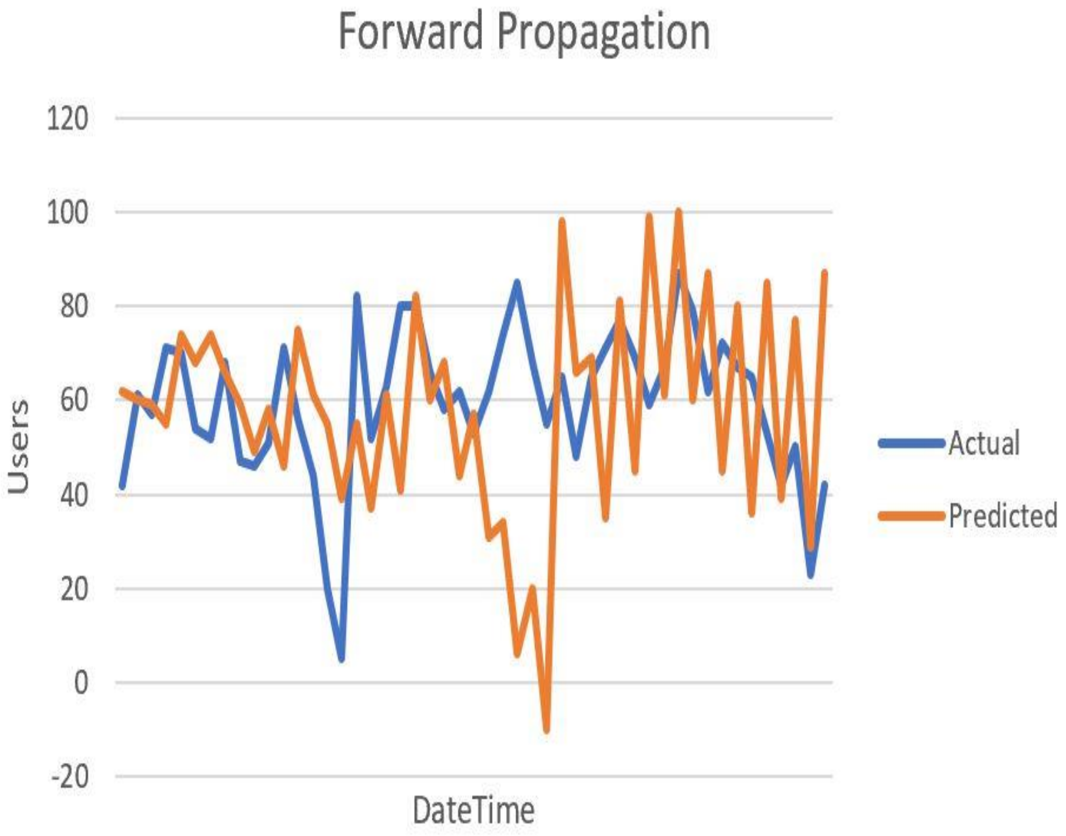


Fig. 21:Actual and predicted values using forward propagation for the NASA Dataset

With the forward propagation, the results were improved by reducing the RMSE. These results could have been further tuned to have more accuracy using the back propagation. We presented a naïve method in which the dataset was divided into train and test randomly, with 75% and 25% respectively, and calculated the predicted load which assumed the previous values. This method produced a result that was then used as the benchmark for the later analysis. Then we used the same dataset with partition as train and test to calculate the predicted load using exponential smoothing. The exponential smoothing method used the smoothing constant and calculated the predicted load using the polynomial expression of 0.6. The results from the exponential smoothing method were improved compared to the naïve approach. The value of smoothing constants determines how fast the weights of time series decays, and values are chosen subjectively or objectively. When the values of a smoothing constant near one put almost all weight on the most recent observations and on the other hand values of a smoothing constant near zero allow the distant past observations to have a large influence. To choose the values subjectively, we can use our experience.

Then popular ARIMA method for time series was used to calculate the load. With ARIMA, the results improved dramatically compared to the other methods. The dataset was used to implement deep learning algorithm and the setup of data was based on the trial and error procedure to find a good number of neurons, m, in hidden layer (HL). ARIMA can be implemented in the predictor layer of the framework suggested in Fig.1 to serve the requests from the end users. Although, there is the possibility of some deviations from the actual and predicted workload, this deep learning model shows the ability to reduce the root mean square error RMSE. The formula to calculate RMSE with which the prediction results can be tested for errors is as follow:

The summary of forecasting and the respective RMSE is calculated and represented in Table 2. RMSE is the standard deviation of prediction that measures the regression line data points.

$$RMSE=\sqrt{\frac{\sum_{i=1}^{N}(Predicted_i - Actual_i)^2}{N}}$$

This experiments gives us an overview of how the time series data is fitting to produce best results and can be used in forecasting, and regression analysis to verify experimental results.

Methods	**RMSE**
Naive Method	68.01
Exponential Smoothing	28.06
ARIMA	25.08
Deep Learning (Forward propagation)	18.98

Table 2: Root mean square error for NASA dataset

The neural mechanism works on the neurons and their corresponding weight applied (Bitsakos, Konstantinou & Koziris 2018). Moreover, with more training data and more neurons added will produce closer results of prediction. Therefore, the proposed framework can be used to predict the workload to ensure a quality service is provided by the cloud providers to the users.

Chapter 7

7. Conclusion

We presented a naive method in which the dataset was divided into train and test randomly, with 75% and 25% respectively, and calculated the predicted load which assumed the previous values. This method produced a result that was then used as the benchmark for the later analysis. Then we used the same dataset with partition as train and test to calculate the predicted load using exponential smoothing. The exponential smoothing method used the smoothing constant and calculated the predicted load using the polynomial expression of 0.6. The results from the exponential smoothing method improved from the naïve approach. The value of smoothing constants determines how fast the weights of time series decay, and values are chosen either subjectively or objectively. When the values of a smoothing constant which is less than one is put in almost all weight on the most recent observations and on the other hand values of a smoothing constant near zero allow the distant past observations to have a large influence. To choose the values subjectively, we can use our experience.

Then popular ARIMA method for time series was used to calculate the load. With ARIMA, the results improved dramatically. The dataset was used to implement deep learning algorithm and the setup of data was based on the trial and error procedure to find a good number of neurons, m, in hidden layer (HL). This can be implemented in the predictor layer of the framework suggested in Fig.1 to serve the requests from the end users. Although, there are possibilities of some deviations from the actual and predicted workload, the deep model shows the ability to reduce the RMSE. The neural mechanism works on the neurons and their corresponding weight applied (Bitsakos, Konstantinou & Koziris 2018). Moreover, with more training data and more neurons added will produce closer results of prediction. Therefore, the proposed framework can be used to predict the workload to ensure a quality service is provided by the cloud providers to the users.

In this thesis, I discussed provisioning of cloud resources using time series and machine learning in the cloud environment. We introduced a dynamic provisioning framework to predict the workload. Using the framework, we calculated the expected load by both time series and machine learning techniques. The model included a predictor layer to predict the load. This layer also analysed the new request and ensure the analysed results were sent to the next layer. This will help in controlling the increasing volume of data through put of enterprises and businesses which are moving towards the cloud computing services. It will also be helpful with analysis of configuration issues on provisioning these resources dynamically, especially when the applications are running during the peak hours. We found that deep learning is efficient in both identifying the load and predicting the workload as well as improving the efficiency of VM. We can also use the above predictions to determine the CPU and memory usages. The future studies will specifically focus on prediction of provisioning vertical or parallel processing of resources by exploring back propagation, RNN and other deep learning algorithms in order to predict efficient usage of the CPU requirements.

To summarise this thesis, auto scaling problems along with workload prediction is provided. It also focusses on the different auto scaling methods that have appeared in the literature review.

7.1 Contribution 1

An effective workload prediction framework with deep learning mechanism is proposed in Fig.4 to address the first research objective. Three major layers were proposed in the framework to identify the workload to the server. The first layer is the load receiver that receives the request from the end users. Then the load analyser layer analyses the load that was received and sends the data to the predictor layer where the prediction of future load to server is calculated.

7.2 Contribution 2

Time series and machine learning techniques are used to experiment with the data obtained from the two different datasets. We introduced the details of the dataset in Sections 4.1 and 4.2. To address the second research question we performed experiments of datasets using linear regression, ARIMA, naïve, exponential smoothing and forward propagation. Determining the workload was considered most important to predict the future loads to the server. Hence, workload prediction is measured as a crucial process in determining the future resources required. This thesis shows that the workload prediction process is achieved in a better way and more efficiently with machine learning. In this thesis, the attributes considered were users and time of request to the servers. These attributes contributed the most in understanding the request send back and forth to the servers. Although many approaches were experimented, the results from forward propagation was considered more accurate compared to the other methods for both the datasets.

7.3 Contribution 3

The third research question is answered with the experiments in Sections 5.1, 5.2, 6.1, 6.2, 6.3. After analysing the various approaches in the literature review and experimenting on the datasets, we can conclude that time series data can be better managed with predicting the workload than any other methods. The thesis shows that the applications which are based on the users input and time should be capable of handling the loads in a better way. Ensuring that the system is not interrupted from providing the resource is crucial to calculating the workload in advance. Furthermore, workload calculation is used in this thesis to improve the efficiency of the cloud servers thus providing resources required dynamically.

7.4 Future work

The aim of this thesis is to be able to create a framework to calculate the workload to cloud servers which in turn can assist in dynamically provisioning the cloud resources. Based on the workload calculated, the number of virtual machines is determined and released. If the workload prediction model works appropriately, then the wastage of resources can also be avoided and this can reduce the cost to cloud service users and cloud service providers. Time series, machine learning and deep learning techniques have proved to be useful and powerful in this research. Although the workload prediction was able to identify the future load, future work is required to determine the number of virtual machines to be released along with identifying the specification of those virtual machines, which will give way to vertical scaling. Also it is important to include the files transferred, the size of the file and the response from the server along with identifying the number of servers used with the information about the server. With understanding this critical information, cloud services can be utilised without any interruptions and more efficiently due to lower associated cost and by not wasting any resource.

Author Publication

Bhagvathiperumal, S., & Goyal, M., 2019. Dynamic Provisioning of Cloud Resources Based on Workload Prediction. In *Computing and Network Sustainability* (pp. 41–49). Singapore: SpringerLink.

Bhagvathiperumal S, Goyal M. Workload Analysis of Cloud Resources using Time Series and Machine Learning Prediction, In 6th IEEE CSDE'19, the Asia Pacific Conference on Computer Science and Data Engineering, Dec 9–11, Melbourne, Australia (in print).

References

Abugaber, D., <https://ademos.people.uic.edu/Chapter23.html>.

Aggarwal, C.C. 2018, 'Neural Networks and Deep Learning', *Springer International Publishing*.

Alipour, H., Liu, Y. & Hamou-Lhadj, A. 2014, 'Analyzing auto-scaling issues in cloud environments', paper presented to the Proceedings of 24th Annual International Conference on Computer Science and Software Engineering, Markham, Ontario, Canada.

Arlitt, M.F. & Williamson, C.L. 1997, 'Internet Web servers: workload characterization and performance implications', *IEEE/ACM Transactions on Networking*, vol. 5, no. 5, pp. 631-45.

Aslanpour, M.S., Ghobaei-Arani, M. & Nadjaran Toosi, A. 2017, 'Auto-scaling web applications in clouds: A cost-aware approach', *Journal of Network and Computer Applications*, vol. 95, pp. 26-41.

Bao, B., Xiang, Y., Li, L., Lyu, S., Munshi, H. & Zhu, H. 2018, 'Scalable Cloud Service For Multimedia Analysis Based on Deep Learning', *2018 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pp. 1-4.

Bitsakos, C., Konstantinou, I. & Koziris, N. 2018, 'DERP: A Deep Reinforcement Learning Cloud System for Elastic Resource Provisioning', *2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pp. 21-9.

Bob Gill, T.B., Chirag Dekate 2019, *Hype Cycle for Edge Computing, 2019*, Gartner, viewed 09/08/2019 2019, <https://www.gartner.com/document/3956137?ref=solrAll&refval=242751505&toggle=1>.

Brownlee, J. 2019, *Deep Learning & Artificial Neural Networks*, Deeplearning, Machine Learning Mastery, Australia, viewed 16/08/2019 2019, <https://machinelearningmastery.com/what-is-deep-learning/>.

Calcavecchia, N.M., Caprarescu, B.A., Di Nitto, E., Dubois, D.J. & Petcu, D. 2012, 'DEPAS: a decentralized probabilistic algorithm for auto-scaling', *Computing. Archives for Informatics and Numerical Computation*, vol. 94, no. 8-10, pp. 701-30.

Cheng, M., Li, J. & Nazarian, S. 2018, 'DRL-cloud: Deep reinforcement learning-based resource provisioning and task scheduling for cloud service providers', *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 129-34.

Choi, J., Ahn, Y., Kim, S., Kim, Y. & Choi, J. 2015, 'VM auto-scaling methods for high throughput computing on hybrid infrastructure', *Cluster Computing*, vol. 18, no. 3, pp. 1063-73.

Daitan 2019, *Exponential Smoothing Methods for Time Series Forecasting*, Better Programming03, Aug 2019, <https://medium.com/better-programming/exponential-smoothing-methods-for-time-series-forecasting-d571005cdf80>.

Durkee, D. 2010, 'Why Cloud Computing Will Never Be Free', *Queue*, vol. 8, no. 4, pp. 20-9.

Dutreilh, X., Rivierre, N., Moreau, A., Malenfant, J. & Truck, I. 2010, 'From data center resource allocation to control theory and back', *Proceedings - 2010 IEEE 3rd International Conference on Cloud Computing, CLOUD 2010*, pp. 410-7.

Frank, E., Trigg, L., Holmes, G. & Witten, I.H. 2000, 'Technical Note: Naive Bayes for Regression', *Machine Learning*, vol. 41, no. 1, pp. 5-25.

Frank, R.J., Davey, N. & Hunt, S.P. 2001, 'Time Series Prediction and Neural Networks', *Journal of Intelligent and Robotic Systems*, vol. 31, no. 1, pp. 91-103.

Ghobaei-Arani, M., Jabbehdari, S. & Pourmina, M.A. 2016, 'An autonomic resource provisioning approach for service-based cloud applications: A hybrid approach', *Future Generation Computer Systems*.

Hassan, M., Chen, H. & Liu, Y. 2018, 'DEARS: A Deep Learning Based Elastic and Automatic Resource Scheduling Framework for Cloud Applications', *2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud*

Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom), pp. 541-8.

Huang, Y., Ma, X., Fan, X., Liu, J. & Gong, W. 2017, 'When deep learning meets edge computing', *2017 IEEE 25th International Conference on Network Protocols (ICNP)*, pp. 1-2.

Kim, H.W. & Jeong, Y.S. 2016, 'Efficient auto-scaling scheme for rapid storage service using many-core of desktop storage virtualization based on IoT', *Neurocomputing*, vol. 209, pp. 67-74.

Liu, C., Shie, M., Lee, Y., Lin, Y. & Lai, K. 2014, 'Vertical/Horizontal Resource Scaling Mechanism for Federated Clouds', *2014 International Conference on Information Science & Applications (ICISA)*, pp. 1-4.

Lorido-Botran, T., Miguel-Alonso, J. & Lozano, J.A. 2014, 'A Review of Auto-scaling Techniques for Elastic Applications in Cloud Environments', *Journal of Grid Computing*, vol. 12, no. 4, pp. 559-92.

Mahmud, A.H., He, Y. & Ren, S. 2015, 'BATS: Budget-Constrained Autoscaling for Cloud Performance Optimization', *2015 IEEE 23rd International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pp. 232-41.

Mao, M. & Humphrey, M. 2011, 'Auto-scaling to minimize cost and meet application deadlines in cloud workflows', *SC '11: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1-12.

Mogouie, K., Mostafa Ghobaei, A. & Shamsi, M. 2015, 'A Novel Approach for Optimization Auto-Scaling in Cloud Computing Environment', *International Journal of Modern Education and Computer Science*, vol. 7, no. 8, pp. 9-16.

Qiu, F., Zhang, B. & Guo, J. 2016, 'A deep learning approach for VM workload prediction in the cloud', *2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pp. 319-24.

Qu, C., Calheiros, R.N. & Buyya, R. 2016, 'A reliable and cost-efficient auto-scaling system for web applications using heterogeneous spot instances', *Journal of Network and Computer Applications*, vol. 65, pp. 167-80.

Roy, N., Dubey, A. & Gokhale, A. 2011, 'Efficient Autoscaling in the Cloud Using Predictive Models for Workload Forecasting', *2011 IEEE 4th International Conference on Cloud Computing*, pp. 500-7.

Sapankevych, N.I. & Sankar, R. 2009, 'Time Series Prediction Using Support Vector Machines: A Survey', *IEEE Computational Intelligence Magazine*, vol. 4, no. 2, pp. 24-38.

Sedaghat, M., Hernandez-Rodriguez, F. & Elmroth, E. 2013, 'A virtual machine re-packing approach to the horizontal vs. vertical elasticity trade-off for cloud autoscaling', *ACM International Conference Proceeding Series*.

Sung, C., Higgins, C.Y., Zhang, B. & Choe, Y. 2017, 'Evaluating deep learning in chum prediction for everything-as-a-service in the cloud', *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 3664-9.

Venters, W. & Whitley, E.A. 2012, 'A critical review of cloud computing: Researching desires and realities', *Journal of Information Technology*, vol. 27, no. 3, pp. 179-97.

Wang, H., Pannereselvam, J., Liu, L., Lu, Y., Zhai, X. & Ali, H. 2019, 'Cloud Workload Analytics for Real-Time Prediction of User Request Patterns', *Proceedings - 20th International Conference on High Performance Computing and Communications, 16th International Conference on Smart City and 4th International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2018*, pp. 1677-84.

Wang, W., Chen, H. & Chen, X. 2012, 'An Availability-Aware Approach to Resource Placement of Dynamic Scaling in Clouds', *2012 IEEE Fifth International Conference on Cloud Computing*, pp. 930-1.

Wikipedia 2019, *Autoscaling*, <https://en.wikipedia.org/wiki/Autoscaling>.

Yan, H., Yu, P. & Long, D. 2019, 'Study on Deep Unsupervised Learning Optimization Algorithm Based on Cloud Computing', *2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*, pp. 679-81.

Yang, J., Liu, C., Shang, Y., Cheng, B., Mao, Z., Liu, C., Niu, L. & Chen, J. 2014, 'A cost-aware auto-scaling approach using the workload prediction in service clouds', *Information Systems Frontiers*, vol. 16, no. 1, pp. 7-18.

Ying-chen, L. 2011, 'The importance of cloud computing to the development of financial informatization', *2011 6th International Conference on Computer Sciences and Convergence Information Technology (ICCIT)*, pp. 968-70.

Zhao, Z., Barijough, K.M. & Gerstlauer, A. 2018, 'DeepThings: Distributed Adaptive Deep Learning Inference on Resource-Constrained IoT Edge Clusters', *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 11, pp. 2348-59.