Addis Ababa University

College of Natural Sciences

Application Service Behavior Prediction Model Over Inter-cloud Environment

Soreti Tilahun Negasa

A Thesis Submitted to the Department of Computer Science in Partial Fulfillment for the Degree of Master of Science in Computer Science

Addis Ababa, Ethiopia
22 March, 2021

Addis Ababa University

College of Natural Sciences

Soreti Tilahun Negasa

Advisor: *Dagmawi Lemma* (PhD)

This is to certify that the thesis prepared by *Soreti Tilahun Negasa*, titled: *Application Service Behavior Prediction Model over Inter-cloud Environment* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

Name	Signature	Date

1. Advisor : Dagmawi Lemma (PhD) ____________________

2. Examiner: ____________________

3. Examiner: ____________________

Abstract

Cloud computing is a computing model that delivers different services to its users through Internet. These services include storage, databases, networking, analytics and software. Nevertheless, the delivery of these services towards the users will be difficult, if resources on the cloud are overloaded due to increased workloads. To overcome this situation an environment called Inter-cloud environment is designed. This environment is designed by forming cloud of clouds where each cloud would use the computational, storage or any kind of infrastructural resource of other clouds. However, the aggregation of diversified computing systems in the Inter-cloud environment poses difficult problems in effective delivery of application services and resource provisioning. These problems arise because of the magnitudes and uncertainties of Inter-cloud components (workload, compute servers, services). This research aims to study the Inter-cloud environment along with the behaviors of the application services and to propose a prediction model that assists the environment with knowledge to future resource surge of each service. The application service behaviors prediction model will be used to predict the CPU utilization of Inter-cloud services. The prediction model was developed by the most widely used machine learning method, Artificial Neural Network (ANN). Among the Artificial Network Algorithms; Multilayer Perceptron Neural Network (MLP) is used to approximate any linear or non-linear function. MLP method is employed to develop application service behavior prediction model.

The Inter-cloud environment is simulated using FederatedCloudsim framework. Materna workload traces and Bitbrain workload traces are used to generate random resource workload traces from the FederatedCloudsim framework. The generated resource workload traces have been used to analyze the problem, to train and test the proposed prediction model. Four experiments were designed to build the application service behavior prediction models using generated resource workload data of Materna workload and Bitbrain workload traces from Inter-cloud environment. From the evaluation of the prediction model two factors that could affect the accuracy of the predicted results are pointed out. In this work the Coefficient of Determination and Mean Squared Error metrics are used to analyze the accuracy of the predictor model.

Key words: *Inter-cloud Environment, Application Service Behavior, ANN, MLP, Machine Learning, Prediction Model.*

Dedication

I would like to dedicate this work to God.

Acknowledgments

First of all, I would like to thank the almighty God for giving me the strength to finish this work. I offer my genuine gratitude to my advisor, Dr. Dagmawi Lemma. He has supported me throughout this work with his patience, motivation, immense knowledge and professional advice. His guidance, encouragement and advice helped me in analyzing the process of this work.

I will use this opportunity to extend my gratefulness to my family as a whole, you all mean a lot to me. My family has given me their support throughout my work, as always. For which my expression of thanks likewise doesn’t suffice. I always thank God the almighty for giving me such a wonderful family. Wish you all a happy long life.

I would like to acknowledge the love, help, support, patience and motivation of my friends. You are the reason for my strength in any difficulties I was suffering throughout this thesis work. Finally, thanks to all my classmates. You all have my sincerest appreciation.

Be Blessed.

Table of Contents

List of Figures iii
List of Tables iv
List of Algorithms v
Acronyms vi
Chapter One: Introduction 1
1.1 Background 1
1.2 Motivation 3
1.3 Statement of the Problem 3
1.4 Objective 5
1.5 Methods 5
1.6 Scope and Limitation 6
1.7 Application of Results 6
1.8 Organization of the Rest of the Thesis 7
Chapter Two: Literature Review 8
2.1 Introduction 8
2.2 Cloud Computing 8
2.3 Virtualization in cloud computing 9
2.4 Need for Resource Provisioning 11
2.5 Resource Provisioning in Cloud Computing 11
2.6 Overview of Data mining 12
2.7 The need for Prediction in Resource Provisioning 13
2.8 Prediction Techniques 14
Chapter Three: Related Work 23
3.1 Resource provisioning for applications in heterogeneous platforms 23
3.2 Resource provisioning for applications services in cloud environment 26
3.3 Conclusion 28
Chapter Four: Design of Application Behavior Prediction Model in Inter-cloud Environment 29
4.1 Inter-Cloud Environment and Need for Interoperability 29
4.2 Components of Service Behavior Prediction Model in Inter-cloud Environment 31
Chapter Five: Implementation and Evaluation 43
5.1 Overview 43
5.2 Development Tools 43
5.3 Implementation Detail 44

5.4 Experiments and Experimental Results....................46
5.5 Experimentation setup....................47
5.6 Result and Discussion....................47
5.7 Summary of Experimental Results....................53
Chapter Six: Conclusion and Future Work....................56
6.1. Conclusion....................56
6.2 Contribution....................57
6.3 Future Work....................57
References....................58
Appendix-1: Sample Code Segment for Prediction Component....................64

List of Figures

Figure 2.1: Multi-layer Artificial Neural Network 16

Figure 4.1: Generic Inter-cloud Environment and service prediction model is applied 30

Figure 5.1: A sample Gephi illustration of VM migration within Inter-cloud environment...............46
Figure 5.2: Mean Squared Error for predictor of 4CSPs with Maternal workload data 48
Figure 5.3: R^2 values for predictor of 4CSPs with Maternal workload data......................................48
Figure 5.4: Scatter graph of Actual Vs Predicted values from Experiment 1 49
Figure 5.5: Mean Squared Error for predictor of 5CSPs with Maternal workload data 50
Figure 5.6: R^2 values for predictor of 5CSPs with Maternal workload data......................................50
Figure 5.7: Scatter graph of Actual Vs Predicted values from Experiment 2 50
Figure 5.8: Mean Squared Error for predictor of 4CSPs with Bitbrain workload data 51
Figure 5.9: R^2 values for predictor of 4CSPs with Bitbrain workload data.......................................51
Figure 5.10: Scatter graph of Actual Vs Predicted values from Experiment 3 52
Figure 5.11**:** Mean Squared Error for predictor of 5CSPs with Bitbrain workload data 52
Figure 5.12: R^2 values for predictor of 5CSPs with Bitbrain workload data.....................................53
Figure 5.13: Scatter graph of Actual Vs Predicted values from Experiment 4 53

List of Tables

Table 4.1: Generated random workload data of Materna workload from federation 4CSPs 36
Table 4.2: Generated random workload data of Materna Workload from federation of 5CSPs 37
Table 4.3: Generated random workload data of Bitbrain workload from federation of 5CSPs 37
Table 4.4: Generated random workload data of Bitbrain workload from federation of 4CSPs 38
Table 4.5: Used ANN parameters 41
Table 5.1: Number of workload data in each experiment……………………………………………54
Table 5.2: Mean and Variance of MSE values from each experiment 55

List of Algorithms

Algorithm 4.1: Algorithm for the FDCB component..33
Algorithm 4.2: Algorithm for the CC component 34
Algorithm 4.3: Algorithm for the Cloud Broker component..35
Algorithm 4.4: Algorithm for Data Preprocessing 39
Algorithm 4.6: Algorithm for the Predictor component 41

Acronyms

ANN	Artificial Neural Network
APA-VMP	Adaptive Power-Aware Virtual Machine Provisioner
AR	Autoregressive
ARMA	Autoregressive Moving Average
AWS	Amazon Web Services
CC	Cloud Coordinator
CPU	Central Processing Unit
CSP	Cloud Service Provider
EAC	Elastic Application Container
FCS	FederatedCloudSim
FDCB	Federated Datacenter Broker
ICRP	Inter-cloud Resource Provisioning
MA	Moving Average
MLP	Multilayer Perceptron
MSE	Mean Squared Error
PM	Physical Machine
QoS	Quality of Service
RBF	Radial Basis Function
RRBF	Recurrent Radial Basis Function
SLA	Service Level Agreement
VM	Virtual Machine

Chapter One: Introduction

1.1 Background

Currently, cloud computing is becoming the platform to provide computing services over the Internet. Cloud computing is a computing model that helps cloud users to access resources in a dynamic and virtualized manner, which are provided as services.

The cloud model has five essential characteristics [1]: on-demand self-service, rapid elasticity, multi-tenancy, measurability of service, and cost-effectiveness.

- On-demand self-service: resources in cloud computing are provided on demand. The request of the cloud user will be processed without the intervention of human on the provider’s side.
- Multi-tenancy: in cloud computing is an architecture where multiple distinct user groups use shared hosted resources.
- Rapid elasticity: Cloud computing offers the capabilities of rapid scaling, up or down according to the current demand.
- Measured service: cloud computing has the capability to control and optimize resource that is appropriate to the type of services.
- Cost-effectiveness: Cloud computing is on demand pay-as-use i.e billing is done based on the usage of the customer which downs the operational and capital cost.

In addition, the cloud computing model can be deployed using one of the four deployment models [2]: these are private cloud, public cloud, community cloud, and hybrid cloud.

- Private cloud: this type of cloud deployment resides within only a single organization which allows them to run their services in isolation from any other organization. The cloud services cannot be used by the public and only the organization has the right to access this cloud.
- Public cloud: this cloud deployment and the virtualization environment is open for public uses. Amazon Web Services (AWS) is a good example of a public cloud.
- Community cloud: the cloud is shared by several parties with common requirements such as privacy, data processing, and jurisdiction. These clouds are managed either internally or by a third-party provider.
- Hybrid cloud: refers to the services hosted among public and private clouds. In this deployment model, the orchestration among the public and private clouds takes place so that the cloud users can use the services provided by the clouds in an exchangeable way.

The aim of cloud computing is to provide computational resources to consumers at a lower cost. In order to allocate resources efficiently, the cloud environment has the ability to deal with the unpredictable projection of resource consumption by the consumers. If computational resources on the cloud are overloaded due to increased workloads, the service towards the consumers will be degraded. Inter-cloud architecture [3] is designed to overcome this situation by forming clouds of clouds, where each cloud may use the computational, storage, or any kind of resource of the infrastructures of other clouds. In [4], Inter-cloud computing is considered as a model with the aim of assuring the quality of services (QoS), such as the performance and availability of each service. This model is developed with the purpose of allowing reassignment of requested resources and the transfer of workloads within different cloud providers.

Buyya *et al.* [5], proposed a federated cloud computing environment, which is an Inter-Cloud environment that facilitates scalable provisioning of application services. Federated cloud is developed to achieve QoS by matching various resource needs under variable workload, and network conditions. Also, cloud computing providers shall be able to predict the geographical distribution of users who are consuming their services. Hence, the load coordination must happen automatically and the distribution of services must change in response to changes in the load [5]. The Inter-cloud computing model envisioned the aggregation of diversified and flexible functionalities of computing systems. Nonetheless, effective provisioning and delivery of application services in the Inter-cloud environment faces difficult problems resulted from the uncertainty and unpredictable resource requirement and effects on its components such as workload and services.

According to [5], provisioning is defined as a “high-level management of computing, network, and storage resources that allow them to effectively provide and deliver services to customers”. Previous attempts were made to predict the performance of other IT systems to enable resource management. Most of these attempts are based on either statistical functions or machine learning algorithms. The authors in [6, 7] worked on forecasting the future behavior of application in terms of workload based on collected information in cloud computing. The work suggest that predicting application service behaviors can be a essential for efficient resource managements in a homogeneous cloud environment. The previous efforts to predict application service in a homogeneous environment haven’t been applied to a heterogeneous characterized environment such as, Inter-cloud environment. To make use of the potential of the Inter-cloud computing model, there is a need for a predictive model as an efficient solution for effectively estimating resource demand of services.

Predictive analysis is the stem of several essential system designs and deployment decisions such as workload management, system sizing, capacity planning, and dynamic rule generation in a cloud environment [6]. This thesis proposes a prediction or forecasting model based on historical data in terms of different resource requirements of services hosted in the Inter-cloud environment. Fortunately, the resources required can be consistently predictable to a very useful degree from historical data. In order to gather the needed historical data for the proposed predictor model from the Inter-cloud environment, a FederatedCloudsim simulator [8] is used. In the deployment of the Inter-cloud environment, two real-time workload traces are used. After applying these workload traces in the Inter-cloud environment a random resource workload is generated and the generated workload trace in terms CPU and memory utilization of the cloud providers host is used in order to train, validate and test the proposed prediction model with a minimum time.

1.2 Motivation

Though cloud infrastructure can provide ample resources to its subscribers, the resource is not as abundant but limited. A cloud environment cannot provide to the cloud users if all the computational resources are exhausted. In such a situation, the cloud running out of resources might obtain the resources by subscribing/collaborating with other cloud infrastructure. This implies that computing services and Internet-based cloud computing technologies are becoming more attractive and form a cloud of clouds or lead to Inter-cloud architecture. It is also necessary that the application service behavior needs to have a prediction model on this architecture to support resource provisioning and dynamic scaling of resources.

In general, currently we are unable to find a prediction model that is developed for application service behaviors over Inter-cloud architecture. Also, the need for standard QoS for the customers of cloud computing is a motivating factor for this work so as to come up with a model that can improve resource provisioning mechanisms.

1.3 Statement of the Problem

Cloud computing deals seamlessly with unexpected rise in resource consumption but if these resources are overloaded due to high resource demand, that is not yet available, then service to the user may degrade. In order to eliminate the degradation of services, it is important to envision effective provisioning of resource/s.

An effort has been made by Islam *et al.* [7] to investigate resource provisioning problems from the cloud platform provider's perspective. Thus, the study [7] uses an intelligent real-time prediction to evaluate subsequent resource (such as, CPU, memory, network, etc.) demand of a single homogeneous cloud provider. The intelligent prediction discussed by [7, 28, 34] does enhance resource provisioning for a single homogeneous cloud provider but still there is a need for further investigation of resource provisioning in a heterogeneous computing system. The cloud stacks of participating cloud providers in the Inter-cloud environment tend to be heterogeneous. Due to the heterogeneous property of the participating clouds in the Inter-cloud environment, interoperability problems will arise. The cloud incompatibilities can be either from physical and virtual implementation, or the software code [9]. The incomparability causes lock-in issues that resulted from the policies used by providers to govern the resource management of the offered resources and services [10]. This challenge will limit the consumer's ability to compare and choose the cloud offerings while switching between cloud providers.

In Inter-cloud environment the selection of appropriate cloud provider that can host the requested services is executed through the component called Cloud Broker (CB). However, the CB would benefit if the future resource serge of the services hosted by the provider can be predicted beforehand. Predicting the future resource demand of the services will assist the task of CB and improve the resource provisioning problem without the struggle of the interoperability problem.

Resource demand prediction can be done by analyzing the behavior of the application services. To enable the provisioning of the physical and virtual resources of the providers in the Inter-cloud environment in a more seamless way, the unpredicted workload fluctuation from one cloud to another needs to be investigated [10]. In our thesis work, the utilized physical and virtual resources of each participating cloud provider in the Inter-cloud environment are measured in terms of CPU and memory usage and used for the prediction of the future resource surge of the services in order to adapt with the change in workload as in [7]. The aim of this study is to develop a model that can predict the behavior of application services so that the future surge in resource requirement can be speculated within inter-cloud environment. And, we hypothesize that the prediction can be based on a collected data using different parameters of the service behaviors from multiple heterogeneous clouds.

1.4 Objective

General Objective

The main objective of this thesis is to design an application service behavior prediction model for Inter-cloud environment.

Specific Objectives

In particular, this thesis has the following specific objectives:

- To Study and identify parameters useful for the prediction.
- To study possible statistical methods that can be applied in the model.
- To study the works and the current practices that have been done in the area of prediction models that supports dynamic resource provisioning of services over the cloud computing model.
- To identify techniques, methods and define equations to give load for the proposed model.
- To design, adopt and develop a suitable approach based on the identified techniques and the defined equation.
- To develop a model that predicts the behavior of application service based on the newly identified approach.
- Conduct experiment based on the proposed model over the Inter-cloud environment.
- Test and evaluate the proposed application service behavior prediction model.

1.5 Methods

We will apply experimental research method within design science framework to develop the model, which will be the main artifact of this investigation. Also, in order to accomplish the general and specific objectives of the study, different methodologies will be employed. Beginning from reviewing relevant literature to acquire a deeper understanding of the research area and its problem domains, the methodology will include designing, implementing and testing the proposed model.

i) Data Collection

For training and testing the prediction model, resource workload data gathered from Inter-cloud environment will be used. The workload data for the predictor model will be gathered by deploying real-time Materna Workload Trace (GWA-T-13_Materna-Workload-Traces) [12] and fastStorage trace from Bitbrains Workload Trace (GWA-T-12 Bitbrains performance Trace) [13] in the Inter-cloud environment.

ii) Experimentation

In order to accomplish the objective of the research, experiment will be carried out by using simulation of the Inter-cloud environment and the prediction model to show the proposed model is not only able to make accurate projections but also skilled enough to forecast resource demand. The experimentation includes historical data collection from the simulation of Inter-cloud environment and training, validating and testing the prediction system with the historical data.

iii) Testing and Evaluation

The accuracy of the prediction model will be assessed using evaluation metrics through simulation and real world data.

1.6 Scope and Limitation

The main focus of this study is developing a model that predicts or forecasts the future resource demand of application service hosted over Inter-cloud environment. This research concentrates on the development of Inter-cloud environment that takes service level agreement (SLAs) into consideration and prediction model that forecast the service behaviors in terms of resource workload. This study will not analyze the implement of the prediction model on the Cloud Broker (CB) component of the Inter-cloud environment. Our work also excludes investigation on further resource provisioning techniques and works.

1.7 Application of Results

The result of this research is believed to be used either as an input for other researches or can be put into use in different fields. The possible applications of the proposed application service behavior model will:

- Be used in predicting the behavior of application services for provisioning of resources and delivery of services in Inter-cloud architecture.
- Open the way for further researches in the area of Inter-cloud computing that handles intelligent management of resources by satisfying the QoS targets for its customers.

1.8 Organization of the Rest of the Thesis

The organization of this thesis work is discussed as follows: In Chapter 2, we review relevant literature on the background of cloud computing, virtualization in cloud computing, need for resource provisioning, resource provisioning in cloud computing, the need for prediction in resource provisioning, and prediction techniques used in time-series prediction.

In Chapter 3, the review of related works is presented. The related works discussed are categorized under resource provisioning for applications in heterogeneous platforms and resource provisioning for applications services in a cloud environment.

In Chapter 4, the design of the proposed architecture is presented. In this chapter, the need for interoperability in the Inter-cloud environment is discussed. During designing the architecture, tasks are classified and basic components of the Inter-cloud environment and prediction model are presented. The different algorithms of each component in the architecture are also discussed.

Chapter 5 presents the implementation and discussion of the proposed work based on the architecture designed in Chapter 4. The cost for training, validating and testing the proposed prediction model is taken into consideration, to generate such good predictions. It also presents four experiments done with the evaluation results. The results from the evaluation of the neural network predictor achieve consistent improvement in predicting future resource workload values. After being trained for only a couple of seconds with all the generated workload data from the Inter-cloud environment, our simple network is able to predict the desired workload values without the need of being trained again. The results show that the proposed model will help the system to be able to predict the behaviors of the hosted services in terms of resource demand, so that it intelligently undertakes decisions related to service/s provided over the Inter-cloud environment.

In Chapter 6, the conclusion and the contributions of the thesis are presented. Furthermore, issues that have been surfaced while working on the thesis are presented as future works.

Chapter Two: Literature Review

2.1 Introduction

This chapter presents a review of the literature to provide a brief background on cloud computing environment, different types of cloud services with the challenges, virtualization in cloud computing, resource provisioning in cloud computing, the need for resource provisioning and Virtual Machine (VM) migration, the techniques engaged for forecasting purposes. Different prediction techniques are applied for resource provisioning in many literatures but this review only gives a brief discussion of time-series prediction models.

2.2 Cloud Computing

The concept of cloud originated when providers begin to use a virtual private network for data communication [14]. Cloud computing is a recent trending in IT that transforms computation and data from desktops and portable PCs into large servers [15]. The feature that differentiates the cloud from the traditional computing paradigm is its seemingly infinite amount of resource capacity offered at a competitive rate. Cloud computing is a large distributed computing paradigm that provides on-demand access, dynamic provisioning, rapid elasticity, massive scalability, and multi-tenancy to a shared resource pool. The main goal of a cloud environment is to utilize distributed resources in a better way and achieve quality service by combining them.

Key characteristics of cloud computing is discussed in [15] as follows:

1) Clouds are distributed geographically, in a manner that is transparent to the consumers.
2) Implements a pool of computing resources and services that are shared amongst consumers.
3) Provisioning and configuration (and de-configuration and unprovisioning) of resources and services happen in a self-service manner, implying that the process occurs automatically with no human operator assistance.
4) The underlying infrastructure is virtual and built on top of the physical resource servers, disks, network segments, etc., facilitates transparent encapsulation of service and resource.
5) The biggest strength of the services is rapid elasticity [16]. Cloud-provisioned resources can have the auto-scaling capability to turn the service on when the load is high or shut itself down when the resource is idle [17].
6) Cloud systems can control and optimize the resources and services using an “as used” metered and/or capacity-based model.

There are three typed of cloud services [18].

1) Software- as-a-Service (SaaS): is cloud service model that provide applications as a service to the cloud customers through Internet. Instead of buying and installing applications users can get the software with less money. In utilizing SaaS hardware requirements for users will be reduced.
2) Platform-as-a-Service (PaaS): is a service model which provide resources which is used for the purpose of building applications completely. In this models developers can design, develop, test and deploy and host applications. When utilizing PaaS the main challenge is compatibility. There is not APIs, software, database, languages or features that are common to all PaaS providers [19]. PaaS is prone to compatibility and vendor lock-in issues.
3) Infrastructure-as-a-service (IaaS): is a model that provide required hardware to cloud users. This service model allows rent resources like server space, cpu cycles, network equipments, memory and storage space. By utilizing this service help the environment to dynamically scaled up or down based on the users request. Particular big challenges of Iaas are intensive computing workload, the use of security policies by the providers and the physical location of data.

2.3 Virtualization in cloud computing

The emerging of cloud computing as a large computing resource pool such as processing power, storage, software, and network resource to the cloud consumers integrates the concept of virtualization as its backbone. Hence, cloud computing provides storage and computing services to its consumers it is difficult to manage the workload using the traditional computing environment [18]. As a result, the concept of virtualization plays a major role to make the cloud computing environment more elastic and efficient. The concept of virtualization is to transform a single machine interface into the illusion of many virtual machines [20]. Each of these virtual machines is an efficient replica of the original computer system.

Virtualization can be applied to a wide range such as operating system virtualization, hardware-level virtualization, storage-level virtualization, and application-level virtualization. The advancement in virtualization improves the scalability of computing resources (software and hardware) in the cloud environment. The other application of virtualization is sharing resources among applications that run in different machines to respond more effectively to the changing in demand. On a single physical machine, multiple VMs can be constructed and execute concurrently to meet the different requirements of service requests using various partitions of resources.

Resource management in a virtualized environment is crucial. Hence, the VM resource demand should be met while the number of Physical Machine (PM) used is minimized. This is challenging when the resource needs of the VMs is occupied considerably due to the diverse set of application they run and the guest operating system for running the applications [21, 22]. If a virtual unit that can run its own instance, it will be much easier to provisioning of cloud resource [22]. Using a mapping of VM for specific services onto a set of server will help to provisioned resource dynamically from the resource pool [23].

In a virtualized environment VM live migration is a widely used technique for dynamic resource allocation [24, 25]. Employing the migration of VMs across physical machines assist to scale up or scale down unused resources depending on the application service requests. Machine virtualization technologies like Xen provide a mechanism for mapping virtual machines (VMs) to physical resources [26]. It is up to the cloud provider to make sure the PMs have sufficient resources to meet cloud users needs. The technologies can be a great deal in resource management.

2.4 Inter-cloud environment

Inter-cloud environment is when multiple cloud providers work together to allow on-demand assignment of resources, transfer of workload, and guarantee service quality [27]. Inter-cloud environment provides has many benefits for both the cloud consumers and cloud providers. The benefits of Inter-cloud environment are discussed as follows:

1) Diverse geographical locations: Leading cloud service providers have datacentres worldwide. However, it is unlikely that any provider will be able to have datacentres in every country and administrative region [5]. Many applications have law requirements as to where data are stored. By utilizing multiple clouds one can gain access to widely distributed resources and provide well-performing services.
2) Better application resilience: the use of multiple providers will allow applications to be fault tolerant. This has been advised by Amazon to its users to design the application by utilizing multiple datacenter.
3) Avoidance of vendor lock-in: Utilizing multiple clouds allow cloud users to easily avoid vendor lock-in by migrating to another provider if one cloud provider changes its policy and strategies.

4) Expand on demand: a participating cloud provider in the Inter-cloud environment can use the resources from other clouds if workload increases beyond its limits [5].

2.4 Need for Resource Provisioning

Managing the cloud environment resources based on the application service request have to be made in order to efficiently make use of the resources. Determining the right amount of resources in a cloud environment is a double-edged sword, over provisioning and under provisioning of resources must be avoided [28]. When the engaged resources are unable to fulfill the demand due to its uncertainty then under provisioning problem may occur and if the reserved resources are more than the actual demand over provisioning problem can occur. To tackle the challenge of determining the right amount of resources in the cloud is by studying and implementing resource provisioning techniques.

The ultimate goal of resource provisioning is to maximize profit from the Cloud Service Provider's Perspective and from the Cloud User's Perspective to reduce cost [29]. Maximize profit by efficiently allocating the resources. When the cloud users request the cloud service provider to make a provision for the resource the instances and requirement of the resources will be identified for a particular application. Many studies have been conducted in the area of computing resource management. And all have come up with good results; hence resource provisioning can be a great deal for high computational performance. Among them some are discussed below:

2.5 Resource Provisioning in Cloud Computing

Resource provisioning in cloud computing means the selection, deployment, and run-time management of software and hardware resources such as CPU, storage, network, etc for ensuring guaranteed performance for applications [29], while these resources are virtually provided to the service subscriber. Due to the complexity of cloud computing resources, provisioning them can be a guaranty to efficiently make use of the resources and meet QoS. By provisioning and allocating the right resources, QoS parameters like availability, throughput, security, response time, reliability, performance, etc. will be achieved.

Based on the need of applications, resource provisioning and allocation can be static or dynamic. Static Provisioning/Dynamic Provisioning and Static/Dynamic Allocation of resources have to be made with the goal is to efficiently make use of the resources and meet QoS. Provisioning takes place when the cloud user requests the cloud service provider on which instances and requirement of resource needed to be provisioned.

In [29] Resource provisioning is classified into three based on the application behavior in terms of the workload/demand of resources.

a) **Static Provisioning**: this type of provisioning is used for applications with consistent workload/demand. In this type of provisioning the provider prepare appropriate resources in advance and offer the service to its consumers.
b) **Dynamic Provisioning**: in this provisioning type migration of VMs takes place in order to encounter the change of application demand. In this type of provisioning the provider will assign more resources as they are needed and remove them when they are not. The customers that are using this provisioning will use pay-per-use method. When this provisioning type is applied in hybrid clouds, it is known as cloud bursting.
c) **User Self-provisioning (Cloud Self-Service)**: this provisioning takes place when a customer buys a resource by paying with a credit card from a cloud service provider.

2.6 Overview of Data mining

The crucial goal of data mining is determining new, useful, and comprehensive patterns in data [30]. Data mining is a useful process in discovering knowledge from huge data. The mining process is more than data analysis which includes classification, clustering, prediction, and association rule discovery. Techniques from different disciplines such as statistics, machine learning, pattern recognition, neural networks, etc, work in integration with the data mining process [31]. These days a number of data mining applications have been successfully implemented in various domains such as fraud detection, web applications, manufacturing, retail, finance, banking, telecommunication, surveillance, teaching assistant, health care, etc.

The application of data mining will fall into three categories: predictive modeling, descriptive modeling and discovering pattern and rules [32], each of them are discussed as follows.

a) Predictive Modeling

In predictive modeling, there are two variables the dependent and the independent variables. The dependent variable has value that is derived from the values of the independent variables. Predictive modeling has a process of predicting future values from a known data pattern. Classification and regression are two forms of data analysis that can be used to predict future data [31]. Classification methods will find a typical pattern and examine classified classes to create a new class. The regression method predicts the future values of the dependent variable based on the historical relationships with other variables. The difference between classification and regression is the type of output that is predicted; classification predicts class membership whereas regression models continuous-valued functions [31].

b) Descriptive Modeling

In descriptive modeling, the distribution of data and groups are studied and modeled using the relationship of the variables from the distributions. Clustering and data visualization are the most known techniques used for descriptive modeling. Clustering describes data by finding a suitable category or a cluster for it. The process in clustering involves the segmentation of data that fall into the same cluster and assigning data points based on similarities with the identified cluster. The goal of clustering is to make a partitioning of the input data from the given data. There are different functions that yield different outputs. It is up to the miner to determine what meaning, if any, to attach to the resulting clusters [16]. The visualization method is another powerful form of descriptive data mining [16]. It presents data in both the input and output steps of the process. The relationship between the properties at the input stage and the decisions presented at the output stage is studied using visualization techniques.

c) Discovering Patterns and Rules

In this type, detecting a pattern is the main concern. Among the most commonly known applications in discovering patterns and rules are handwritten recognition (OCR) and face recognition [33].

2.7 The need for Prediction in Resource Provisioning

An accurate prediction methodology that estimates the resource demand based on historical data is essential to make use of the optimal resources in a cloud environment in an efficient way. Mapping VMs to PMs in the next interval a head needs a predictor to forecasts resource demand in the future observation interval [25]. Building a predictor require analysis of historical resource usage data. Resource prediction model in the cloud will make hosted applications to withstand the variation in workload with the least drop in performance and availability [8]. Resource prediction model can also be used instead of traditional single prediction metrics for VM provisioning and meeting SLA requirements. Employing the prediction model not only helps the cloud system to meet SLA requirements but it also gives the cloud client a more robust scaling decision choice [34].

2.8 Prediction Techniques

A number of existing techniques to make an acceptable prediction of future application service behavior in terms of resource requirement in the cloud with respect to time are explored here. These techniques are able to predict the most likely future outcome based on recent resource usage and historical data. Hence, the research problem actually relates to time-series analysis. Some of the most common and popular methodologies in time-series analysis are discussed below.

A time series is a set of measurements of variable values at various times. Depending on the frequency of the time series data different patterns emerge in the data which form the component to be modeled. At times, the time series set may encounter a constant increase or decrease in values. In which case, if the series across time is purely random in nature it is known as white noise. In this series, a pattern is not indicated hence it is difficult to forecast future values. Such series have a mean value of zero, constant variance, and an uncorrelated random variable. There are several different approaches to time series forecasting, which are generally categorized as linear and non-linear models [30]. Among both categories, we are going to discuss some of the models which are widely used in the time series forecasting process.

i) Linear stationary models

In [35] Box and Jenkins discussed three linear stationary models: Autoregressive model (AR), Moving Average model (MA) and Autoregressive Moving Average model (ARMA). In an AR model, one variable only depends on its own past values. The autoregressive process Y_t depends on Y_{t-1}, Y_{t-2}, Y_{t-3},.., *etc*. Thus, $Y_t=f\,(Y_{t-1}\,,\,Y_{t-2}\,,\,Y_{t-3}\,,\,\dots\,,\mathcal{E}_t)$. The model is a simple linear regression model in which a sample value is predicted based on the previous sample values. The autoregressive process Y_t of a system that is based on the previous values is given by the formula:

$$\boldsymbol{Y_t = a_1 Y_{t-1} + a_2 Y_{t-2} + a_3 Y_{t-3} \ldots + \mathcal{E}_t} \qquad (1)$$

where, a_1, a_2 and so on are the coefficients determined by the auto-correlation coefficients of the time series which are used to determine whether the model is stationary or not. $\mathcal{E}_t$ is some noise term and the autoregressive process Y_t is a non-negative integer.

The second type of linear model is moving average model, Y_t depends only on the random error terms which follow a white noise process, i.e, $Y_t=f(\mathcal{E}_t\ ,\ \mathcal{E}_{t-1}\ ,\ \mathcal{E}_{t-2}\ ,\ \mathcal{E}_{t-3}\ ,\ \ldots\ ,,)$. A common representation of MA model is given in Equation 2.

$$Y_t = \beta_0 + \varepsilon_t + \phi_1 \varepsilon_{t-1} + \phi_2 \varepsilon_{t-2} + \phi_3 \varepsilon_{t-3} + \ldots + \phi_q \varepsilon_{t-q} \tag{2}$$

where the error terms $\mathcal{E}_t$ are assumed to be white noise processes with mean zero and constant variance.

There are situations where the time series may be represented as a mix of both AR and MA models referred as ARMA. The data value at any given time t, say y_t , is considered as a function of the previous data values say Y_{t-1} , Y_{t-2} , $Y_{t-3,}$ and the errors at times t, t − 1,..., t − q, say n_t, n_{t-1} , . . ., n_{t-q}. The general form of such a time-series model takes the form:

$$y_t = a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_p y_{t-p} + n_t + b_1 n_{t-1} + \cdots + b_q n_{t-q} \tag{3}$$

where, a_1 to a_p are the autoregressive coefficients and b_1 to b_q are the moving average coefficients.

So, using the three linear stationary models in the appropriate model building process, several iterative steps of model identification, parameter estimation, and diagnostic checking should be applied until a stationary model is finally selected. These linear stationary models are not flexible when it comes to a broad range of non-linear problems.

ii) The ANN approach to time series forecasting

ANN is a computational intelligence system developed to mimic biological neural networks [39]. In recent decades, ANNs have shown great ability in modeling and forecasting nonlinear and non-stationary time series in computing. Some of the advantages of ANNs are:

1) The ANNs are able to recognize the relation between the input and output variables without stated physical consideration.
2) Even when the training sets contain noise the ANNs will work without errors and are able to approximate any continuous function to any desired accuracy [26, 30].
3) The ANNs are able to adapt to solutions overtime to compensate for changing circumstances and once trained they are easy to use.

This is a nonlinear modeling technique that is suitable for modeling over a very wide range of applications [30]. It is more flexible in terms of architecture. The neural-network architecture bears high similarity to the neurons in the brain, hence the name “artificial neural network.” In this network architecture, there may be two or more layers. For example, a three-layer ANN has three layers, namely an input layer, a hidden layer, and an output layer. The inputs can be of any number. Also, the number of neurons in the hidden layer is flexible. ANN model is applied in the field of prediction because of its inherent capability of arbitrary input–output mapping. Each node in the hidden and output layers receive input from the previous nodes, process them locally through an activation function, and produce an output to the next nodes. The network can learn the data with a learning algorithm, and form a mapping between inputs and the desired outputs from the training set through learning. After the learning process has finished the network can understand the hidden dependencies between the input and outputs. This implies that the network has the capability of nonlinear generalizability [37]. To build a neural network model for prediction, the process should pass through three stages:

1) The training stage where the network is trained to predict future data, based on past and present data.
2) The testing stage where the network is tested to stop training or to keep in training.
3) The evaluation stage where the network quit training and is used to forecast future data and to calculate different measures of error.

In the ANN model, three layers of networked units connected by acyclic links are presented. Figure 3.1 [38] illustrate example of a three layer ANN.

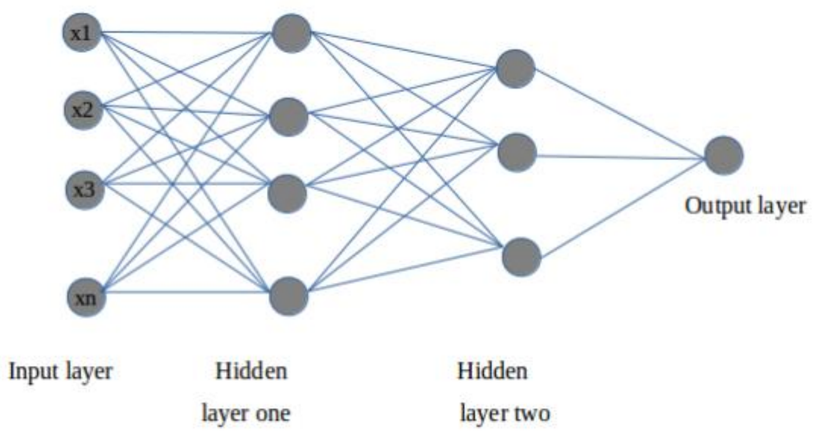


Figure 2.1: *Multi-layer Artificial Neural Network*

The network depicted in Figure 3.1 has 4 network inputs where external information is received in our case access pattern is the input, and 1 output layer with one node where the solution is obtained. The network inputs and the output layer are separated by 2 hidden layers: the first layer with 4 nodes, and a second layer with 3 nodes. The connections between the nodes indicate the flow of information from one node to the next, i.e from left to right. Each node has the same number of inputs as the number of nodes in the preceding layer and each connection is modified by a weight. Also, each node has an extra input assumed to have a constant value of 1 which is called the bias. Equation (4) shows the relationship between the inputs ($y_{t-1},\ldots\ldots,y_{t-p}$) and output ($y_t$) of the network in a mathematical equations.

$$y_t = w_0 + \sum_{j=1}^{q} w_j \cdot g\left(w_{0j} + \sum_{i=1}^{p} w_{i,j} \cdot y_{t-i}\right) + \varepsilon_t, \tag{4}$$

where, $w_{i,j}$ and w_j are called connection weights; p is the number of input nodes, and q is the number of hidden nodes.

The simple network given by Equation (4) is a powerful tool that can approximate the number of hidden nodes when q is large. Practically, a simple network structure can work well for a sample data predictions process if it has a small number of hidden nodes. However, this might lead to poor generalization to the data out of sample. The choice of q is data-dependent and there is no systematic rule in deciding this parameter. The most important tasks in building the ANN network are choosing the appropriate number of hidden nodes and selecting the number of input observations. For determining these parameters there is no simple method and the usual procedure is to test different network structures [39]. After the network structure is identified, the network can be trained for future uses. Estimating the needed parameters will minimize the cost function of a neural network. The cost function is a criterion used to measure the accuracy of the network such as Mean Squared Error (MSE) [40].

When the network is running, each node in the hidden layer and output layer perform the calculation in the form of Equation (5) on it’s input, and transfer the result O_c to the next layer.

$$O_c = h\left(\sum_{i=1}^{n} x_{c,i} w_{c,i} + b_c\right)$$
$$\text{where} \quad h(x) = \begin{cases} \dfrac{1}{1+e^{-x}} & \text{if hidden layer node} \\ x & \text{if output layer node} \end{cases} \tag{5}$$

where, O_c is the output of the current node, n is the number of nodes in the previous layer, $x_{c,i}$

is an input to the current node from the previous layer, $w_{c,i}$ is the weight modifying the corresponding connection from $x_{c,i}$, and b_c is the bias.

Activation functions can take several forms. Hence, the role of the activation function is to transfer inputs to the hidden layer, it is not used in the input layer. In the output layer, a linear function is widely used as an activation function. This is because non-linear activation functions may present a damaged predicted value. Equations (6) and (7) are called a logistic and hyperbolic functions respectively, which are often used in the hidden layer.

$$Sig(x) = \frac{1}{1 + exp(-x)}, \tag{6}$$

$$Tanh(x) = \frac{1 - exp(-2x)}{1 + exp(-2x)}. \tag{7}$$

For ANN to make meaningful predictions, the neural network needs to be trained on an appropriate dataset. Basically, training is a process of determining the connection weights in the network. During the training phase, the Neural Network is fed with input vectors and random weights are assigned to the links. After presentation of each input vector, the network generates a predicted output. The generated output is then compared with the actual output; the difference between the two is known as the error term which is then used as a feedback to correct the weights of the network. Since the error value guides the network towards convergence to the target output for a set of real-world inputs. The rate at which the weight is updated is called the learning rate, which is generally a value within the range [0, 1]. The training of the Neural Network continues until a specific criterion is met, e.g., the sum of squared errors falls below a certain threshold. The overall algorithm for training ANN for prediction can be expressed as the following:

Step 1: Initialization of weights

- Initialize weights to some random value

Step 2: Feedforward

- Take the inputs and propagate them to the upper layers
- For an input vector x_i , compute Equation (5) until the network output activation values are found.

Step 3: Backpropagation of errors

- Compute the error term for the output layer nodes, i.e., the difference between the desired output (the output vector) and the actual network output.

$$e_p = d_c - O_c \quad (8)$$

Where, d_c is the desired output and O_c is the actual output

- hen these errors are propagated sequentially backward from right to left to calculate errors for hidden layer nodes based on the following equations

$$e_p = O_c (1 - O_c) \sum_{n}^{i=1} e_i w_{i,c} \quad (9)$$

where, O_c is the output of the current hidden layer node, n is the number of nodes in the next layer, e_i is the error for a node in the next layer and wi,c is the weight modifying the corresponding connection from the current node to that node.

Step 4: Updating the weights and the bias

- For each connection, the change in the weight modifying the connection from one node to the other is computed using Equation (10) and added to the weight.

$$\Delta w_{c,p} = \dot{\alpha}\, e_p\, O_m \quad (10)$$

where, $\dot{\alpha}$ is is the learning rate of the network, e_p is the error of node p and O_m is the output of node m. The learning rate controls how quickly and how finely the network converges to a solution.

Step 5: Repeat step 2-4 until a specific termination criterion has been met, i.e, the total prediction error is lower than a specific limit.

The above algorithm follows a gradient-descend technique [37] to reach the local optima. Thus, the resultant neural network guarantees to produce an optimal prediction output after observing the input vector in real time.

Evaluating the goodness of the neural network after building is crucial to know the performance of the model. The performances of the ANN models can be evaluated by evaluation metrics: coefficient of determination (R^2) and Mean Squared Error (MSE). In almost all prediction problems the performance of a model is measured by R^2 and MSE [24, 6].

a) Mean Squared Error

Mean Squared Error (MSE) is the average of the squared values of the prediction errors or squared values of the deviations from the target [40]. MSE is the average squared error between the network outputs and the target. Generally, MSE can evaluate and compare the predictive power of the neural network; how close are the predicted values to the target. If the predictor neural network estimations is accurate, then the value of MSE must be low. The developed prediction model will be tested using the test datasets. The mean squared error (MSE) estimated over n samples is defined by Equation (11) [40].

$$MSE(\mathrm{p}, \mathrm{p}') = \frac{1}{\mathrm{nsamples}} + \sum_{n=0}^{\mathrm{nsamples}} (\mathrm{pi} - \mathrm{pi}')^2 \tag{11}$$

b) The Coefficient of Determination

The Coefficient of Determination (R^2) provides a measure of how well future samples are likely to be predicted by the neural network. R^2 describes the proportion of variance of the dependent variable explained by the predictor model [41]. If *y* the predicted value of the *i-th* sample and mean of the predicted value is the corresponding true value, then R^2 is computed by Equation 12 [41].

$$R^2 = 1 - \frac{SS_{\mathrm{res}}}{SS_{\mathrm{tot}}} \tag{12}$$

where, The total sum of the squares is given by ,

$$SS_{\mathrm{tot}} = \sum_i (y_i - \bar{y})^2 \tag{13}$$

The error sum of the squares given by ,

$$SS_{\mathrm{res}} = \sum_i (y_i - f_i)^2 = \sum_i e_i^2 \tag{14}$$

R^2 value ranges from zero to one, with zero indicating a random relationship and one indicating perfect prediction; higher R^2 values indicate better prediction. In most cases R^2 value greater than 0.8 is taken as good accuracy result.

iii) Fuzzy time-series

Zadeh [42] introduced the Fuzzy Set Theory that provides a powerful framework to cope up with vague or ambiguous problems and issues. Time-series models had failed to consider the application of this theory until fuzzy time-series was defined by Song and Chissom in [43, 44]. Fuzzy time-series predictive models can be advantages if there is a situation with incomplete data and fewer observations. However, the weakness of fuzzy time-series is that its performance is not always satisfactory.

iv) Hybrid models

In literature, different combination techniques have been proposed in order to overcome the deficiencies of single models and yield more accurate results. Hybrid models can be homogeneous, such as using differently configured neural networks (all Multi-Layer Perceptron), or heterogeneous, such as with both linear and nonlinear models [45]. The basic idea of a multi-model approach is the use of each component model's unique capability to better capture different patterns in the data. Both theoretical and empirical findings have suggested that combining different models can be an effective way to improve the predictive performance of each individual model, especially when the models in the ensemble are quite different [46, 47]. In addition, since it is difficult to completely know the characteristics of the data in a real problem, hybrid methodology that has both linear and nonlinear modeling capabilities can be a good strategy for practical use [45]. In a cooperative modular combination, the aim is to combine models to build a complete picture from a number of partial solutions.

In recent years, more hybrid forecasting models have been proposed and applied to time series forecasting with good prediction performance. In [48] a hybrid methodology to exploit the unique strength of heterogeneous models is proposed and used. By using a hybrid approach the limitations of the models to capture the behavior of the data will be tackled. The researchers also assert that the hybrid models will improve the prediction performance and accuracy. Meanwhile, compared with the single forecasting method, the hybrid modeling method has a higher forecasting precision to complex problems.

In summary, Inter-cloud systems have been extensively studied with the aim of providing unlimited resources to satisfy maximum customer requirements. However, resource provisioning has become a challenge hence the organization of each participating cloud in the Inter-cloud environment has heterogeneous policies and description of various resources. With an efficient resource provisioning system, an Inter-cloud environment can provide guaranteed application services to its users. In this chapter, the need for resource provisioning and various prediction models that have been extensively studied are reviewed. Devising an accurate prediction model can be a great way to make use of the optimal resources of the clouds and to provide the system with robust decision making. Various studies have shown different linear stationary models and time series models can be a great deal to be used for forecasting in cloud environment.

Chapter Three: Related Work

The subject of many studies becomes managing resources in a heterogeneous computing environment from the application service providers and the cloud service provider's point of view. Hence, several techniques have been studied to make use of the resources in a cloud environment efficiently. Among these techniques, prediction analysis is applied in most of the research we reviewed. In this Chapter, from the solutions for resource provisioning in a computing systems with heterogeneous resource, the following are described as related work to this thesis work. We reviewed studies that have been discussed mainly on systems that have heterogeneous characteristics because our study focuses on solving resource provisioning issues in a multiple cloud environment with heterogeneous cloud stacks. For better understanding, we categorized them as resource provisioning for applications in a computing systems with heterogeneous characteristics, and resource provisioning in a cloud environment.

3.1 Resource provisioning for applications in heterogeneous platforms

The authors in [31] presented a resource provisioning technique for service providers to allocate on-demand compute, network, and storage resources. However, managing the resources using utility systems from a heterogeneous resource pool is the challenging task that has been discussed so far by the paper. By identifying the performance behavior of the hosted applications can help the system to manage its resources more effectively. The researchers presented an active learning approach to build a reasonably accurate predictive model by analyzing performance histories of frequently used applications; the histories consist of measures gathered from noninvasive instrumentation of resources. The predictive model is built by combining aprior structure and statistical learning techniques.

Automatic on-line process behavior extraction, classification, and prediction model for heterogeneous distributed computing systems is proposed in [49]. Distributed systems
The main aim of this work is to have knowledge about application behavior. Hence, the performance of distributed computing is limited due to the nature of heterogeneous distributed systems data transfer rate access latency. In order to efficiently optimize the heterogeneous distributed system performance the knowledge of application behavior is essential. The researchers represent the application behavior by its execution pattern (access pattern). So, the intent of the research was to study new techniques for access pattern discovery and prediction based on neural network and statistical approach. Based on the application behavior extraction the authors proposed

two types of prediction techniques. The first one calculates the incidence matrix to generate Markov Chains for a statistical approach that allows defining the next state based on the current one. All application state transitions are included by calculating the probability of each state transition. This allows predicting all states that can be reached by the application at any given execution point and the probability of accessing them. The second one evaluates the application execution history as time series data for prediction using neural networks. This method predicts the future application state changes based on the variations of the application behavior during execution. This model allows efficient prediction of consecutive application state transitions by allowing a larger prefetching window. By both techniques, it is possible to predict when and how many resources would be required to process future operations. Such information can be applied by schedulers to minimize the total application execution time. This study predicts the application behaviour after extraction and classification of access pattern is applied. The uses of sliding window restrict the access pattern length. The use of sliding window will limit the model to the specific patterns and prevents the model from learning the other useful patterns. Also, choosing the access pattern length is should be selected in a way that most popular patterns are extracted.

Evgueni and Rodrigo in [50], proposed a novel approach to predict application behavior. The authors put a correct extraction of process behavior for predicting its future operations. The main goal of this work was to develop a resource provisioning approach based on the prediction of application behavior. The knowledge of the application behavior is essential to make assumptions on future data access and process state changes. After the extraction of application behavior, it can be represented by a sequence of state changes or time series. Such series are also known as application access patterns, and represent the sequence of events performed by an application (such as disk, I/O, CPU activities, network transmission, and so on). This paper introduce a dynamic online process behavior prediction model through Radial Basis Function (RBF) and Recurrent Radial Basis Function (RRBF) neural networks. The prediction model is developed for distributed processes. First, for analysis purposes, the process behavior is converted into time series. Based on the time series information how the sequence of events repeats overtime is calculated. Then the similarity among the time series patterns is determined using the false-nearest neighbor algorithm. The result of this is used to determine the required number of nodes of the neural network. The network is fed with the training data, resulting in a number of centers that adequately represents all patterns which are further used to configure the RBF network.

Inter-cloud resource provisioning system (IRPS) that schedules and allocates resources based on ontology is proposed in [51]. In an Inter-cloud environment, the difference in the management policies and the description of various resources and applications in each cloud organization made it difficult to provide the right resources from various cloud providers. So representing a cloud computing environment using ontology can capture the same attributes among the resources [53]. In this system, a metadata repository is used to store the semantic description of cloud providers' resources that are participating in the Inter-cloud environment. This model provides a framework that schedules the customer requests based on Service Level Agreement (SLA) and state of Service Provider (SP) resources cloud customers by using an inference engine algorithm. The overall system assigns resources to the customers' requests based on their priority. The inference engine algorithm will provide the optimal solution for maximizing the effectiveness of resource usage. The main goal of this work is to tackle the interoperability issue raised in the heterogeneous inter-cloud environment and it focuses only on working on the compatibility of each cloud provider based on the SLA agreement.

Buuya [5], proposed InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services. The problem that is addressed by the study is from the cloud provider's point of view. The geographic distributions of the service users of each provider are unknown. The providers need to predict these distributions in order for the load coordination to happen for the different services. The study solve this issue by creating a federated cloud environment (InterCloud) that assists the scalable provisioning of application services while achieving QoS targets. The overall goal of the researchers is to create a computing environment that has unlimited resources and capability to handle variations in service demands. This research puts a new framework for cloud service providers that can dynamically expand or resize their provisioning capability based on sudden spikes in workload. This framework is market-driven and is aware of SLA based services. The researchers tried to see Infrastructure-as-a-Service (Iaas) challenges and the requirement of application scaling. Real-world elastic applications such as social networking and gaming portals, business applications, media content delivery, and scientific workflows have users distributed all over the world. Hence, at different locations and at any time load spike can take place. As a result, this study tries to put an automatic scaling scheme in order to handle the unpredictable seasonal and geographical changes in system workload. To support this scheme and meet the QoS expectations of all cloud application service consumers the Software-as-a-Service (SaaS) providers need to make use of the services from multiple cloud infrastructure service providers who can provide better support for their specific customer needs. To seamlessly provision the services across different cloud providers, a mechanism is needed for the federation cloud environment. At last, the researchers concluded that the proposed framework will offer powerful capabilities to address both services and resources management, but their end-to-end combination aims to dramatically improve the effective usage, management, and administration of cloud systems. This study tries to solve the load coordination and sudden workload spikes of the cloud services by using the service users geographical distribution.

3.2 Resource provisioning for applications services in cloud environment

Gong [7] proposed PRedictive Elastic ReSource Scaling for cloud systems (PRESS) to address the problem of assigning the right amount of computing resources that need to be given to the applications running in the cloud. In this approach, both under and over-provisioning problems are addressed by allocating the application resource dynamically since to determine the amount of resource the application needs is non-trivial and if the resource needs are predicted in advance the allocation can be adjusted ahead of the needs. To address this challenge a lightweight elastic resource allocation scheme is developed for cloud service providers. This work is done without the requirement of application profiling, model calibration, or deep understanding of the application.

The main goal of the study is to allocate just enough resources, minimize resource waste, and to track dynamic resource requirements of applications. The researchers use CPU usage to manage the applications resource allocation by setting the CPU resource limit for the virtual machine the application is running in (this is called scaling). The attempt of the proposed approach is to allocate just enough resources to applications to avoid Service Level Objectives (SLO) violations and minimize resource waste. The dynamic resource requirement of applications is continuously tracked and resource demands are predicted using two complementary techniques to do so. PRESS first employs signature-driven resource demand prediction by using signal processing techniques. These techniques identify the historic resource usage of the application called repeating patterns or signatures. Then the signatures are used for predictions. The patterns are caused by repeating requests or iterative computations. If no signature is discovered, a statistical state-driven approach is used to capture the patterns in resource demand. When the resource consumption patterns change prediction models are will be updated. Both signature-driven and state-driven approach will be integrated to make runtime elastic resource scaling. When the application starts, PRESS assumes zero knowledge about the application, and then the resource limit is set to maximum (For Example, 100% of CPU). After a few resource demand samples (For Example, 10) have been acquired, PRESS starts to make predictions. Once it is confident in the prediction, PRESS makes scaling resource allocation. PRESS gives higher priority to avoid under-estimation than over-estimation. Under-estimating the application resource demand will make the scaling system set a resource cover that is lower than the real demand. This will cause SLO violations and also affect the accuracy of the future prediction of resource demand.

In [6], the researchers discussed the problem of adaptive resource provisioning in the cloud from the applications provider's viewpoint. Empirical prediction models are used in order to make the hosted applications capable of autonomic scaling decisions by evaluating their future resource utilization in real-time and the request for additional virtual instance in advance. Managing the resources using a dynamic and proactive resource scaling system based on prediction analysis is proposed by the researchers. As a result, the system might be able to scale proportionally with resource demand or sudden traffic flow. Thus, this turns out to avoid application performance degradation and application unavailability. The necessity of a scaling system that can cope with the unsteady resource usage patterns of applications is the main focus of the researchers. Therefore, resource prediction models are used for facilitating proactive scaling in the cloud so that hosted applications are able to withstand the variation in workload with the least drop in performance and availability. In order to train and test the prediction model, a historical data of e-commerce applications is used. In this paper, several machine learning algorithms have been evaluated with varying sliding window sizes to provide accurate predictions. The researchers concluded that integrating an accurate prediction model with an auto-scaling system will enhance the effectiveness of adaptive resource provisioning in the cloud in terms of performance and cost.

In [23], a lightweight approach called Elastic Application Container (EAC) is proposed as a virtual resource unit for provisioning cloud resources and scalable cloud applications. The proposed virtual resource unit has its own resource capacity (such as number of CPUs, size of RAM) to run an instance of an application. From an internal infrastructure EAC can run the instance of an application without an operating system. Auto-scaling of application is supported by the proposed EAC approach. The auto-scaling feature allows the EAC to track and adjust resource utilization by cloud users. This allows cloud users to concentrate on their work without considering the application behaviors in terms of resources.

Islam et.al [8] discussed an empirical prediction models for adaptive resource provisioning in the cloud computing. The real challenge to devise an intelligent way towards dynamic provisioning of resources in the cloud which is effective in terms of both application cost and performance is addressed by the authors. As a result, a resource prediction models are used for resource provisioning in the cloud so that hosted applications are able to withstand the variation in workload with least drop in performance and availability. In order to train and test the prediction model historical data of e-commerce applications is used. In this paper, several machine learning algorithms have been evaluated with a varying sliding window size to provide accurate prediction. The researchers concluded that utilizing accurate prediction model enhance the effectiveness of adaptive resource provisioning in the cloud in terms of performance and cost.

Li et al. [53] proposed CloudProphet tool. CloudProphet predicts the performance of application by recording the workload trace of the application from the local cloud. Two phases are included in this tool. The first is tracing which is the workload information is recorded from the local run of the application. The second is replaying in which the the traced workload of the application is emulated. The performance emulation determines the prediction of the real performance of the application. However, it can create overhead if the application has many synchronization events.

3.3 Conclusion

In this chapter, different works that have been done concerning resource provisioning for application services in cloud computing and in the heterogeneous platforms are reviewed. The main goal of those works is to allocate just enough resources, minimize resource waste, and to track dynamic resource requirements of applications. Also, studies on the Inter-cloud environment from the cloud provider’s perspective are reviewed. The intensive work that has been done in Inter-cloud environment was only to work on the compatibility of the participating cloud providers based on SLA agreement. The gaps we are trying to solve that haven’t been done in previous studies are resource provisioning and interoperability issues that occur in multiple cloud environments (Inter-cloud). The resource provisioning issues in cloud environment, computing systems with heterogeneous characteristics have been solved in many ways. However, resource provisioning issue gets bigger, when the environment has multiple cloud providers with interoperability problem. When those two issues in Inter-cloud environment occurs the service delivery to the consumers will be difficult. In our study, the main goal is to solve these two bigger challenges that the Inter-cloud environment is facing.

Chapter Four: Design of Application Behavior Prediction Model in Inter-cloud Environment

In this Chapter, the proposed prediction model for Inter-cloud environment is presented. The need for interoperability of Inter-cloud environment is also discussed. Finally, we present the prediction model and the Inter-cloud environment components with their detailed functionality and algorithms. The techniques and algorithm used to develop the proposed prediction model in an Inter-cloud environment is also discussed.

4.1 Inter-Cloud Environment and Need for Interoperability

Service interoperability of either from the same service model (eg,PaaS-PaaS, SaaS-SaaS) or different delivery models (eg, PaaS-IaaS, SaaS-PaaS) on different cloud providers is called Inter-cloud interoperability [54]. This allows the interoperability between cloud services offer cloud users to use their data and workload across multiple cloud providers. Figure 4.1 shows the how the interoperability issue between multiple cloud providers is addressed on the Iaas level by using brokering mechanism.

In developing a prediction model to facilitate effective and efficient resource provisioning for Inter-cloud environment, the interoperability should be the main focus in this thesis work. Since Inter-cloud demands cooperation between heterogeneous computing environments, interoperability problem should be addressed. Solving the interoperability problem must include the management of Inter-cloud components and guaranteed end-to-end services to meet the users' requirements [5]. So, in our work, the proposed prediction model takes the following functionalities into consideration [53, 55, 56, 57]:

- The unity of the participating clouds heterogeneous computing environment as a single resource leasing abstraction.
- Predicting the future application service behaviors based on the historical workload data gathered from the participating clouds in the environment.
- Information about the clouds current resource usage and the application services resource demand will be up-to-date.
- How to provide user interaction with the Inter-cloud environment using a broker system to make the system aware of the resource requirements of the application services.

➢ A system that makes easy integration of heterogeneous environment between the clouds for flexible and reliable access to the resources based on emerging virtualization technologies.

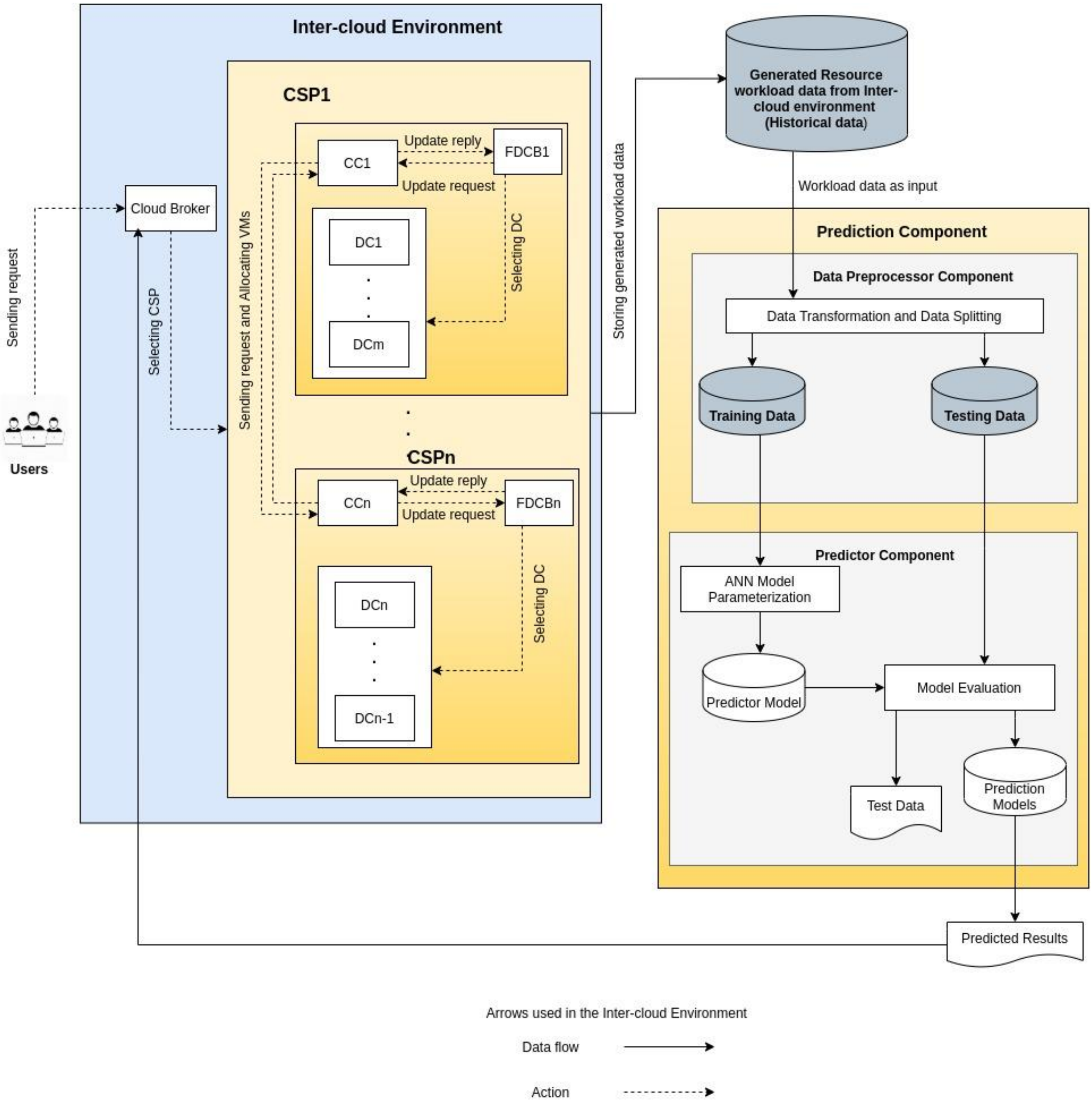


Figure 4.1: *Generic Inter-cloud Environment where application service behavior prediction model is applied*

4.2 Components of Application Service Behavior Prediction Model in Inter-cloud Environment

This section gives a brief description to the proposed prediction model. The proposed model comprises two main parts: the first part shows the integration of two or more cloud service providers to work and access their resources in a collaborative manner and the second part shows the prediction of the application services behavior based on the historical data gathered from the Inter-cloud environment. The illustration of the proposed model for the study is classified into two major components. These are the Inter-cloud environment and the Prediction component.

As shown in Figure 4.1, the Inter-cloud environment consists of *n* number of cloud service providers (CSP) ranging from CSP_1 to CSP_n and each cloud service provider contains *n* number of Datacenters (DC) ranging from DC_1 to Dc_n. The depictions of the cloud service provider as *n* number is used to show that the Inter-cloud environment must be an environment with two or more cloud service providers as its component, where *n* cannot be less than two. Each cloud service provider will contain one or more DC, the n number of DC cannot be less than one.

In this work, identical operational process as that of FederatedCloudSim, a SLA-aware Federated Cloud Simulation Framework [10] is used for the proposed Inter-cloud environment. The working flow for the service request is either sent from a virtual user or migrated from other service providers. When the service request is sent from the Cloud Broker, the Federated Datacenter Broker (FDCB) will estimate whether the task can be run in the local cloud or not. Depending on the available resources and on its strategy, it may ask the Cloud Coordinator (CC) to send requests to the CC of other CSPs and decide based on their offers whether the task is to be run locally or transferred to another cloud. Regardless of whether the CSP accepts a job from the user or from another CSP, the service request is translated into a VM and locally scheduled. In the first step of the scheduling process, the FDCB chooses a DC of the cloud. Each CSP has its own set of brokering and scheduling strategies. The strategies for the CC and the FDCB can consider the workloads of the local data centers, the Service Level Agreement (SLA) between user and CSPs, and the SLAs concluded with other CSPs. After the service behavior data is gathered from the Inter-cloud environment data preprocessing process will be undertaken.

The prediction component illustrated in Figure 4.1 is where the service behavior in terms of resource workload data is preprocessed in order to be used by the predictor model. In the data preprocessing stage the resource workload data will be normalized to appropriate range of numbers and split into training and testing sets. The training set will be used in the process of constructing the prediction model and the testing set will be used to evaluate the prediction model. For the predictor model, a multi-layer feedforward network [59] with a backpropagation training algorithm [60] is chosen to train and test the predictor component. In order to build the model a set of parameters (number of input nodes, number of hidden layers, and number of output nodes) should be identified. After building the model it is trained using the training dataset and training algorithm. Then the model will be tested to know if we have to stop training or to keep in training. Then at last the network will be evaluated where the network is used to forecast future service behaviors and calculate the different errors. Then at last stage the predicted results will be used by the Cloud Broker component to make intelligent decision when choosing the appropriate CSP that can host the requested service.

4.2.1 The Inter-cloud Environment

Our study proposes a prediction model for the behaviors of services in the Inter-cloud environment. Figure 4.1 depicts the main components of the Inter-cloud environment, specifically underlying the Cloud Service Providers (CSP), the Cloud Broker, the Cloud Coordinator (CC), and the Federated Datacenter Broker (FDCB).

4.2.1.1 The Cloud Service Provider Component (CSP)

A cloud service provider component is a major part of the Inter-cloud environment. The cloud service provider component consists of one or more Federated DC, a CC component, and the FDCB component. The CC component help the CSP to communicate with other CSP that are participating in the Inter-cloud environment. The FDCB component will be the one to facilitate the task of selecting the appropriate DC that can host the service within the local CSP. Each CSP in the Inter-cloud environment has its own set of strategies for the brokering and scheduling process. Hence, each CSP will offer their resources in a federated manner and provide the service for the user with respect to SLA requirements.

a) The Federated Datacenter Broker (FDCB) Component

The FDCB is the component that chooses the appropriate federated datacenter from the cloud to carry out the task. The FDCB receives the service request from the Cloud Broker and estimates if the task can run locally. Depending on the available resources and on its strategy, it may ask the CC to send requests to the CC of other CSPs and decide based on their offers whether the task is to be run locally or transferred to another cloud. This component also allocates virtual machines to the Cloud nodes based on the user's requirement. When the service request reaches the CSPs, depending on the available resources of the cloud and on its strategies, it may ask the CC to send the request to other CSPs and decides based on their offer to either run the request locally or transfer it to other CSPs. If it decides the request to run locally, the FDCB will choose the Federated Datacenter based on the scheduling strategies the CSP is using. The scheduler will include SLA to decide whether the services request can run locally.

Algorithm 4.1: *Algorithm for the FDCB component*

```
 Input: service_request, DC, SLA , CSP, acknowledgment_event,
        rejected_service_request
Task: Scheduling the service request to be executed to the appropriate
      DC and allocating VMs within the local cloud
      Start
        Read service request, SLA, DC, CSP;
      if (DC!= null)
         then Create VM
         VM ←service request
          if (service request && SLA == CSP)
           then  service_request_accepted_list ←service_request
          else
              CC ←acknowledgment_event
       else
          CC ←rejected_service_request
     End
```

b) The Cloud Coordinator (CC) Component

The CC component is responsible for managing the status of the clouds, their membership, and their interaction in the overall inter-cloud environment. It provides programming, management, and deployment environment for applications in the Inter-cloud environment [5]. The authors in [5] use CC in a federated cloud environment to export the service of the clouds to the federated environment. The CC we use aid the local cloud to offer their services and resources to the other cloud providers in the Inter-cloud environment. In this study, the CC is responsible for identifying the appropriate Cloud Service Provider to schedule a service request. If a service request that is either from local or remote CSPs is scheduled to be executed locally then the Cloud Coordinator makes contact with FDCB and makes a list of the VMs that are allocated locally. But, if the service request is rejected locally the CC sends a request to the CCs of other CSPs until a CSP comes forward with the required resource to host the requested service. This task may help the FDCB to identify the appropriate decision on whether the tasks can run locally or to transfer it to other CSPs. The other importance of this task is that it will help the decision of VM migration or rescheduling the VM to other CSPs.

Algorithm 4.2: *Algorithm for the CC component*

```
Input: service_request, SLA, localCSP, foreignCSP
 Task:  Scheduling Service Request and allocating the VM in
        the remote cloud
     Start
         Read service_request, SLA
         if(localCSP == null)
           then foreignCSP ←service_request
             if(SLA && service_request == foreignCSP )
                then
                service_requet_accepted_list←service_request
                foreignCSP ←VM
         else
             localCSP ←VM
      End
```

4.2.1.2 The Cloud Broker Component

A cloud broker is defined as a party that negotiates between a cloud service provider and customers [12]. This party has the responsibility to engage in support of the activities of either the providers or the customer or both [58]. The term cloud broker has been used with different meanings in many literatures. In most cases, the cloud broker provides a service that acts on behalf of the cloud service consumers and provision resources [5, 59, 60]. Based on this general idea a cloud broker service will be defined as an entity with the following responsibilities:

- Allocating and deallocating resources for a given application across multiple clouds.
- Use provisioned resources to deploy an application component.

- Implement scheduling and load balancing strategies on the user requests to the allocated resources.

Hence, we are working on a separate cloud infrastructures and we need a broker that make use of the clouds service in aggregation. Cloud brokers that act as an aggregator provide a platform that brings together multiple cloud service provider services and offers them in a federated manner. In our case the cloud service customers (users) select the suitable cloud service provider with the help of the Cloud broker. Cloud brokers accept service request, solicit offers from several CSPs and chooses the most suitable one that can be the one that offers the best guarantees described as a Service Level Agreement (SLA). The predicted results in terms of resource workload from the predictor model can be used by the CB to assist the task of choosing the appropriate CSP.

Algorithm 4.3: *Algorithm for the Cloud Broker component*

```
Input: service_request from (Users), CSP, foreignCSP
Task:  Selecting the appropriate Cloud Service Provider
    Start
       Read  service_request
       if (CSP != null && SLA == true)
        then
           CSP ←service_request
         else
           while (foreignCSP != null && SLA == true)
          then
              foreignCSP ←service_request
     End
```

4.2.2 The Prediction Component

The function of the Prediction component is to forecast the behavior of application services in the Inter-cloud environment. The prediction component receives the historical resource workload data of the application services from the Inter-cloud environment as input and preprocesses the data for the purpose of training and testing the predictor. The other sub-component of the prediction component will use the preprocessed data and forecast the future application service behavior. The prediction component has two sub-components: The Preprocessor component and the Predictor Component.

4.2.2.1 The Preprocessor Component

Before using the output data of the Inter-cloud environment to train and test the predictor component, it should be preprocessed. In this study, sample data extracted from the Inter-cloud environment will be used. The resource workload data should be preprocessed after it is gathered from the Inter-cloud environment. The historical dataset generated from Inter-cloud environment contains resource workload data from Materna and Bitbrain workload traces. For example, Table 4.1, 4.2, 4.3, and 4.4 shows sample resource workload data generated from the federation of clouds. From both the Materna workload trace and Bitbrain workload trace, we have four datasets for each of the federation of clouds.

Table 4.1: *Generated random workload data of Materna workload traces from federation 4CSPs*

Resource usage logfile ...

time	csp-id	dc-id	host-id	#VMs	total-MHz	used-MHz	ram-conf	ram-used	ram-oc	ram-free	ramPerc	cpuPerc
301	3	8	1	5	93632	12572	44490	7615	0.0347578125	1272385	0.0059492188	0.1342703349
301	3	8	3	5	93632	288	12551	380	0.0098054688	1279620	0.000296875	0.0030758715
301	3	8	5	5	93632	440	54668	402	0.042709375	1279598	0.0003140625	0.0046992481
301	3	8	7	5	93632	10827	73903	11520	0.0577367188	1268480	0.009	0.1156335441
301	3	8	9	5	93632	3199	79719	2626	0.0622804688	1277374	0.0020515625	0.0341656699
301	3	8	11	5	93632	22203	55295	15279	0.0431992188	1264721	0.0119367188	0.2371304682
301	3	8	13	5	93632	3243	61397	2388	0.0479664063	1277612	0.001865625	0.0346355947
301	3	8	15	5	93632	433	43620	459	0.034078125	1279541	0.0003585938	0.0046244874
301	3	8	17	5	93632	13891	57694	5786	0.0450734375	1274214	0.0045203125	0.1483573992
301	3	8	19	5	93632	433	55345	566	0.0432382813	1279434	0.0004421875	0.0046244874
301	3	8	21	5	93632	3661	50786	4089	0.0396765625	1275911	0.0031945313	0.0390998804
301	3	8	23	5	93632	2414	98139	4572	0.0766710938	1275428	0.003571875	0.025781784
301	3	8	25	5	93632	420	77888	758	0.06085	1279242	0.0005921875	0.0044856459
301	3	8	27	5	93632	13244	46448	8232	0.0362875	1271768	0.00643125	0.1414473684
301	3	8	29	5	93632	6039	40117	5807	0.0313414063	1274193	0.0045367188	0.0644971805
301	3	8	31	5	93632	3322	12704	933	0.009925	1279067	0.0007289063	0.0354793233
301	3	8	33	5	93632	706	28596	2415	0.022340625	1277585	0.0018867188	0.0075401572
301	3	8	35	5	93632	1043	288096	507	0.225075	1279493	0.0003960938	0.0111393541
301	3	8	37	5	93632	386	9390	226	0.0073359375	1279774	0.0001765625	0.0041225222
301	3	8	39	5	93632	1214	167594	542	0.1309328125	1279458	0.0004234375	0.0129656528
301	3	8	41	5	93632	136	19516	351	0.015246875	1279649	0.0002742188	0.0014524949
301	3	8	43	5	93632	189	34584	493	0.02701875	1279507	0.0003851563	0.0020185407
301	3	8	45	5	93632	2472	40479	4054	0.0316242188	1275946	0.0031671875	0.0264012303
301	3	8	47	5	93632	20775	69350	10621	0.0541796875	1269379	0.0082976563	0.221879272
301	3	8	49	5	93632	301	39441	590	0.0308132813	1279410	0.0004609375	0.0032147129

Table 4.2: *Generated random workload data of Materna Workload Traces from federation of 5CSPs*

Resource usage logfile ...

time	csp-id	dc-id	host-id	#VMs	total-MHz	used-MHz	ram-conf	ram-used	ram-oc	ram-free	ramPerc	cpuPerc
301	3	8	1	8	93632	1424	50176	3469	0.0392	1276531	0.0027101563	0.0152084757
301	3	8	3	8	93632	2799	52224	1693	0.0408	1278307	0.0013226563	0.0298936261
301	3	8	5	8	93632	1494	49152	5360	0.0384	1274640	0.0041875	0.0159560834
301	3	8	7	8	93632	3277	102400	5052	0.08	1274948	0.003946875	0.0349987184
301	3	8	9	8	93632	12258	38912	4753	0.0304	1275247	0.0037132813	0.1309167806
301	3	8	11	8	93632	5777	53248	11596	0.0416	1268404	0.009059375	0.0616989918
301	3	8	13	8	93632	1331	38912	6005	0.0304	1273995	0.0046914063	0.0142152256
301	3	8	15	8	93632	3567	69632	5031	0.0544	1274969	0.0039304688	0.0380959501
301	3	8	17	8	93632	3606	81920	13691	0.064	1266309	0.0106960938	0.0385124744
301	3	8	19	8	93632	1632	112640	9350	0.088	1270650	0.0073046875	0.0174299385
301	3	8	21	7	93632	2954	34816	6757	0.0272	1273243	0.0052789063	0.0315490431
301	3	8	23	7	93632	1444	188416	2624	0.1472	1277376	0.00205	0.0154220779
301	3	8	25	7	93632	4125	47104	3050	0.0368	1276950	0.0023828125	0.0440554511
301	3	8	27	7	93632	344	26624	2173	0.0208	1277827	0.0016976563	0.0036739576
301	3	8	29	7	93632	8193	88064	9930	0.0688	1270070	0.0077578125	0.087502136
301	3	8	31	7	93632	1425	45056	5247	0.0352	1274753	0.0040992188	0.0152191558
301	3	8	33	7	93632	608	40960	1350	0.032	1278650	0.0010546875	0.0064935065
301	3	8	35	7	93632	2267	77824	4984	0.0608	1275016	0.00389375	0.0242118079
301	3	8	37	7	93632	1065	30720	2313	0.024	1277687	0.0018070313	0.0113743165
301	3	8	39	7	93632	4710	69632	18077	0.0544	1261923	0.0141226563	0.0503033151
301	3	8	41	7	93632	5420	43008	10387	0.0336	1269613	0.0081148438	0.0578861928
301	3	8	43	7	93632	779	38912	1324	0.0304	1278676	0.001034375	0.0083198052
301	3	8	45	7	93632	3890	43008	9832	0.0336	1270168	0.00768125	0.0415456254
301	3	8	47	7	93632	742	45056	1831	0.0352	1278169	0.0014304688	0.0079246411
301	3	8	49	7	93632	6110	61440	8848	0.048	1271152	0.0069125	0.0652554682

Table 4.3: *Generated random workload data of Bitbrain workload data from federation of 5CSPs*

Resource usage logfile ...

time	csp-id	dc-id	host-id	#VMs	total-MHz	used-MHz	ram-conf	ram-used	ram-oc	ram-free	ramPerc	cpuPerc
301	3	8	1	5	93632	12826	81973	7961	0.0640414063	1272039	0.0062195313	0.136983083
301	3	8	3	5	93632	420	42390	405	0.0331171875	1279595	0.0003164063	0.004485646
301	3	8	5	5	93632	691	92340	1057	0.072140625	1278943	0.0008257813	0.007379956
301	3	8	7	5	93632	343	67130	413	0.0524453125	1279587	0.0003226563	0.003663278
301	3	8	9	5	93632	2852	26918	851	0.0210296875	1279149	0.0006648438	0.030459672
301	3	8	11	5	93632	21909	53340	14935	0.041671875	1265065	0.0116679688	0.233990516
301	3	8	13	5	93632	8995	54895	4439	0.0428867188	1275561	0.0034679688	0.096067584
301	3	8	15	5	93632	1129	35145	2794	0.0274570313	1277206	0.0021828125	0.012057843
301	3	8	17	5	93632	15222	129224	10270	0.10095625	1269730	0.0080234375	0.162572625
301	3	8	19	5	93632	3585	47229	7853	0.0368976563	1272147	0.0061351563	0.038288192
301	3	8	21	5	93632	11775	124654	6636	0.0973859375	1273364	0.005184375	0.125758288
301	3	8	23	5	93632	2460	38947	4837	0.0304273438	1275163	0.0037789063	0.026273069
301	3	8	25	5	93632	1753	59544	7011	0.04651875	1272989	0.0054773438	0.018722232
301	3	8	27	5	93632	11736	55226	4980	0.0431453125	1275020	0.003890625	0.125341763
301	3	8	29	5	93632	14963	26513	935	0.0207132813	1279065	0.0007304688	0.159806476
301	3	8	31	5	93632	3100	39236	904	0.030653125	1279096	0.00070625	0.033108339
301	3	8	33	5	93632	3363	35102	6310	0.0274234375	1273690	0.0049296875	0.035917208
301	3	8	35	5	93632	476	14454	815	0.0112921875	1279185	0.0006367188	0.005083732
301	3	8	37	5	93632	521	16408	1124	0.01281875	1278876	0.000878125	0.005564337
301	3	8	39	5	93632	18944	78762	13251	0.0615328125	1266749	0.0103523438	0.202323992
301	3	8	41	5	93632	372	38830	584	0.0303359375	1279416	0.00045625	0.003973001
301	3	8	43	5	93632	582	28778	763	0.0224828125	1279237	0.0005960938	0.006215824
301	3	8	45	5	93632	3229	24735	3686	0.0193242188	1276314	0.0028796875	0.034486073
301	3	8	47	5	93632	4080	169694	3485	0.1325734375	1276515	0.0027226563	0.043574846
301	3	8	49	5	93632	2210	26582	3781	0.0207671875	1276219	0.0029539063	0.023603042

Table 4.4: *Generated random workload data of Bitbrain workload data from federation of 4CSPs*

Resource usage logfile ...

time	csp-id	dc-id	host-id	#VMs	total-MHz	used-MHz	ram-conf	ram-used	ram-oc	ram-free	ramPerc	cpuPerc
301	3	8	1	3	93632	266	11264	495	0.0088	1279505	0.0003867188	0.0028409091
301	3	8	3	3	93632	3821	57344	6686	0.0448	1273314	0.0052234375	0.0408086979
301	3	8	5	3	93632	363	18432	1345	0.0144	1278655	0.0010507813	0.0038768797
301	3	8	7	3	93632	160	16384	461	0.0128	1279539	0.0003601563	0.0017088175
301	3	8	9	3	93632	137	12288	267	0.0096	1279733	0.0002085938	0.001463175
301	3	8	11	3	93632	1529	12288	1116	0.0096	1278884	0.000871875	0.0163298872
301	3	8	13	3	93632	1194	30720	4462	0.024	1275538	0.0034859375	0.0127520506
301	3	8	15	3	93632	146	28672	465	0.0224	1279535	0.0003632813	0.001559296
301	3	8	17	3	93632	900	10240	1277	0.008	1278723	0.0009976563	0.0096120984
301	3	8	19	3	93632	4992	22528	5349	0.0176	1274651	0.0041789063	0.0533151059
301	3	8	21	3	93632	1923	30720	3106	0.024	1276894	0.0024265625	0.0205378503
301	3	8	23	3	93632	808	14336	1232	0.0112	1278768	0.0009625	0.0086295284
301	3	8	25	3	93632	352	151552	392	0.1184	1279608	0.00030625	0.0037593985
301	3	8	27	3	93632	292	13312	1257	0.0104	1278743	0.0009820313	0.0031185919
301	3	8	29	3	93632	1381	20480	231	0.016	1279769	0.0001804688	0.014749231
301	3	8	31	3	93632	1787	69632	4137	0.0544	1275863	0.0032320313	0.0190853554
301	3	8	33	3	93632	223	18432	477	0.0144	1279523	0.0003726563	0.0023816644
301	3	8	35	3	93632	282	34816	1826	0.0272	1278174	0.0014265625	0.0030117908
301	3	8	37	3	93632	302	20480	2004	0.016	1277996	0.001565625	0.003225393
301	3	8	39	3	93632	3245	26624	15880	0.0208	1264120	0.01240625	0.0346569549
301	3	8	41	3	93632	1032	32768	1902	0.0256	1278098	0.0014859375	0.0110218729
301	3	8	43	3	93632	12154	22528	1759	0.0176	1278241	0.0013742188	0.1298060492
301	3	8	45	3	93632	1072	14336	2090	0.0112	1277910	0.0016328125	0.0114490772
301	3	8	47	3	93632	816	38912	3372	0.0304	1276628	0.002634375	0.0087149692
301	3	8	49	3	93632	6653	24576	3265	0.0192	1276735	0.0025507813	0.0710547676

The first step in preprocessing the resource workload data will be selecting the relevant attributes to be used for the training, testing, and validating the predictor component. For example, the sample data for the experimentation from the Inter-cloud environment has many attributes including time, Cloud Service Provider-id (CSP-id), Datacenter-id (dc-id), Host-id (host-id), number of VM (#VM), total-MHz of the host, used-MHz of the host, ram-configured: the capacity of RAM configured in terms of KB, ram-used: capacity of used RAM in terms of KB, ram-oc : (RAM Overclock) in terms of MHZ , ram-free: the capacity of free RAM in terms of KB, ram-Percentage: the used RAM in terms of Percentage and CPU-Percentage: the used CPU in terms of percentage. It shows that we can find as much information as possible about the resource workload behavior of the service from the sample data. The predictor component does not need all of these data and it should be preprocessed before used for prediction. Only the relevant information about the workload of the services is extracted and fed into the predictor component. Preprocessing doesn't always mean removing unnecessary attributes but it also means that the sample data with necessary attributes having null value should be known or removed. Therefore, in this work irrelevant attributes that do not contribute to the prediction will be discarded. So from the overall data gathered from the Inter-cloud environment the following columns values are selected: total-MHz of the host, used-MHz of the host, ram-configured: the capacity of RAM configured in terms of KB, ram-used: capacity of used RAM in terms of KB, ram-oc : (RAM Overclock) in terms of MHZ , ram-free: the capacity of free RAM in terms of KB, ram-Percentage: the used RAM in terms of Percentage and CPU-Percentage: the used CPU in terms of percentage. After extracting the relevant attributes from the sample data, the second step will transform and splitting the workload into the appropriate format for model building.

a) Data Transformation and Data Splitting

Then the next step will be splitting the data we have into a training set and test set. This helps to determine the accuracy of the model after the training is done and see how well it will perform with data that it has not trained on. For instance, Test_size = 0.2 means that 20% of the dataset will be used for testing, and the other 80% will be used for training. In this study, 70% of the data set is used for training and they are store as training data, 10% is used to validate the network while it is training and the remaining 20% are used to test the goodness of the model and stored as test data. After splitting the workload data, transforming the data into a form that is appropriate for model building will be the next step. In this study, data normalization is used to make the dataset appropriate for the neural network tool. For normalizing the datasets Min-Max normalization is used. By using this technique the input attribute values and target attribute value used as training and testing data were normalized into [-1, 1] values.

Algorithm 4.4: *Algorithm for Data Preprocessing*

```
Input: resource_workload _dataset, independent values, dependent
       value
Task: Data transformation and data splitting
    Start
 Read resource_workload _dataset, independent values, dependent value;
     X ←independent values
     Y ←dependent value
     trainingset == 80% (resource_workload _dataset)
     testingset == 20% (resource_workload _dataset)
     X_train == transform(trainingset)
     X_test == transform(testingset)
     End
```

4.2.2.2 The Predictor Component

The function of the predictor sub-component is to estimate the future application service behaviors that are hosted in the Inter-cloud environment in terms of resource workload. The predictor sub-component receives the preprocessed historical resource workload data of the application services and forecast future service behaviors. This sub-component is responsible for predicting the behavior of application services using an Artificial Neural Network (ANN).

a) ANN model Parametrization

Now that the data preprocessing part is done, the next step is to create the structure of our ANN. At this stage of the work, choosing the type of network, the number of nodes in the hidden layer, the learning algorithm, and the activation functions is done. First the number of input nodes in the input layer is determined by the number inputs attributes from the resource workload data used. Then number of hidden layers will be determined. To get a more accurate generalization with fewer weights we chose two hidden layers. To determine the number of nodes in the hidden layer first we use the least number of neuron and evaluate the performance. we then keep increasing the number of hidden layer neurons until optimum result is obtained. At last, to get a higher quality output from the network a single output neuron is used. In this study, the normalized CPU utilization of the application services is used as the target predicted value. Finally, a four-layered MLP network with two hidden layers will be used to construct the MLP models. In this study, to prevent the outputs of the network from reaching very large values reLU (rectifier linear unit) activation function is used in the hidden, and the sigmoid function is used in the output layers. The selected activation functions for this study are widely applicable for prediction problems and resulted in good prediction. The ANN parameters used for constructing the predictor model in this study are listed in the Table 4.5. In this work, a multi-layer feedforward network [38] with a backpropagation training algorithm is chosen to predict the service behaviors hosted in the Inter-cloud environment. This model has been applied in the field of prediction, because of its inherent capability of arbitrary input-output mapping. To build the neural network model the first step will be to import the training set which serves as the input layer. Then, we forward propagate the data from the input layer through the hidden layer to the output layer, where the predicted value is presented. After that, the error between the predicted value and the real value is measured. Then backpropagate the error and use gradient descent [37] to modify the weights of the connections.

Table 4.5: *Used ANN parameters*

Parameter	Values
Input neurons	7
Network layer	4
Number of hidden layer	2
Number of hidden layer neurons	20 and 10 or 30 and 10
Transfer function	RELU ; input to hidden layer and Sigmoid from the hidden layer to the output layer
Training Algorithm	Backpropagation algorithm
Output neuron	1
maximum epochs (iterations)	1,000
Error goal	0
Evaluation metrics	MSE and R^2

Algorithm 4.5: *Algorithm for the Predictor component*

```
Input:preprocessed resource workload data,lr, epochs, units,
      activation, intializer, traning_set, validation_set;
Task: Predict the future resource workload of application services
    Start
      Read preprocessed resource workload data
      Initialize ANN
  Hidden layer ←units=30 && activation==relu && initializer== uniform
  Hidden layer ←units=10 && activation==relu
  Outputlayer ←units=1 && activation==liner && initializer== uniform
  Compile(optimizer == Adam(lr=0.00)
  prediction_history==training_set, validation_set, epochs=1000;
    End
```

After the prediction model is built the next process is to evaluate the goodness of the models. In order to evaluate the performance of the model a testing resource workload dataset is used. In this study, the performances of the models are evaluated by the coefficient of determination (R^2) and Mean Squared Error (MSE). Hence, in most of the prediction problems, R^2 and MSE are utilized for network performance evaluation purposes. So, MSE is used to evaluate and compare the predictive power of the model and help to identify how close the predicted values to the desired output are. If the MSE value approximate to zero, then the prediction model estimation is accurate. Equation (11) shows how MSE values are computed. R^2 is used to evaluate how well the predictor model can generalize any given future sample. To compute the R^2 value of a prediction performance Equation (12) is given. If the value of R^2 approximates to 1, this indicates better prediction. At the last stage of the evaluation models with large R^2 values and small MSE are selected and stored as a prediction model knowledge base to predict the future behaviors of services in terms of CPU utilization workload. After the prediction performance is evaluated using MSE and R^2 values, the predictor model will be good to go. At last, the predicted results from the predictor model will be used by the CB component to assist the selection of appropriate CSP.

In summary this Chapter discusses the design of prediction model for services that are hosted over Inter-cloud environment. First, Inter-cloud environment is developed with CSP, CB, CC, and FDCB components. Second, the predictor model construction, evaluation and prediction is designed. Resource workload data generated from Inter-cloud environment is stored and used for the purpose of training, testing and validating the predictor model. The services behaviour hosted on the Inter-cloud environment will be measured in terms of resource workload trace. Before using this workload trace the data should be preprocessed using attribute selection, data transformation and data splitting. When the data is converted to the appropriate format, it will be divided in to training and testing set. Using the training set and set of parameters the prediction model will be constructed. Then a testing set will be used to evaluate the model by using evaluation metrics. The last output of the predictor model will be used by the CB component. The CB will be able to select the appropriate CSP intelligently by using the prediction results.

Chapter Five: Implementation and Evaluation

In this Chapter, the implementation and evaluation of the proposed prediction model in the Inter-cloud environment are presented. In the first section, we will discuss the implementation detail by describing each component of the proposed architecture and how it is implemented. Then, the evaluation of our approach in predicting service behaviors is presented.

5.1 Overview

The implementation process of the prediction model in the Inter-cloud environment shows future service behavior in terms of the workload of the system. The process starts by getting the required inputs from the Inter-cloud environment about the application service behaviors in terms of the workload of the system. These parameters are gathered from the Inter-cloud environment and stored in the historical data file for the prediction purpose. The implementation includes

- Deploying an Inter-cloud environment with its full components and functionalities,
- Running a Materna workload trace [12] and Bitbrians workload trace [13] gathered from different service range in the deployed Inter-cloud environment and find new output data of the behavior of the service in terms of resource workload.
- Preprocessing the output data from the Inter-cloud environment.
- Utilize the preprocessed data to train, validate and test the neural network using a Back-propagation training algorithm,
- Predicting the behaviors of application services using the trained MLP neural network.

5.2 Development Tools

Several tools and technologies were selected and utilized during the development of the proposed application service prediction model in the Inter-cloud environment. The following are a list of tools and frameworks for programming, extending, training, validating, running, data managing, and operating environment which are used in the implementation process.

FederatedCloudsim 2.0 [8]

FederatedCloudSim (FCS) is a simulation framework that is built as an extension to CloudSim [64]. Research on cloud federations in the real world is very complex and expensive as distributed hardware and software scenarios are needed. Therefore, we used FederatedCloudSim to simulate many federated cloud scenarios while respecting SLAs (service level agreements). Hence, FCS is built in a modular way so that it can be easily extended.

Keras Library [61]
Keras is a neural network library written in Python that is high-level in nature that runs Tensorflow as its backend. In this study, Keras library is used to train, test, validate, and run a neural network. In this work, Keras is used for implementing supervised learning to train, validate, and run a multi-layer perceptron network (MLP) with a backpropagation training algorithm.

Eclipse platform

Eclipse platform is an integrated development environment (IDE). This platform is used for computer programming with an extensible plug-in system for customizing the environment. Eclipse platform is primarily used for developing an application using Java. In this work Eclipse platform is used to run FederatedCloudSim framework 2.96 [63] which is a Java based simulator used to depict our Inter-cloud environment.

Jupyter Notebook [63]

The Jupyter Notebook is a web application used to create code, equations, and visualizations. Its application includes data mining, machine learning, and much more. In this work, we use Jupyter Notebook to run the data preprocessing and the MLP neural network that we are developing.

Data Traces

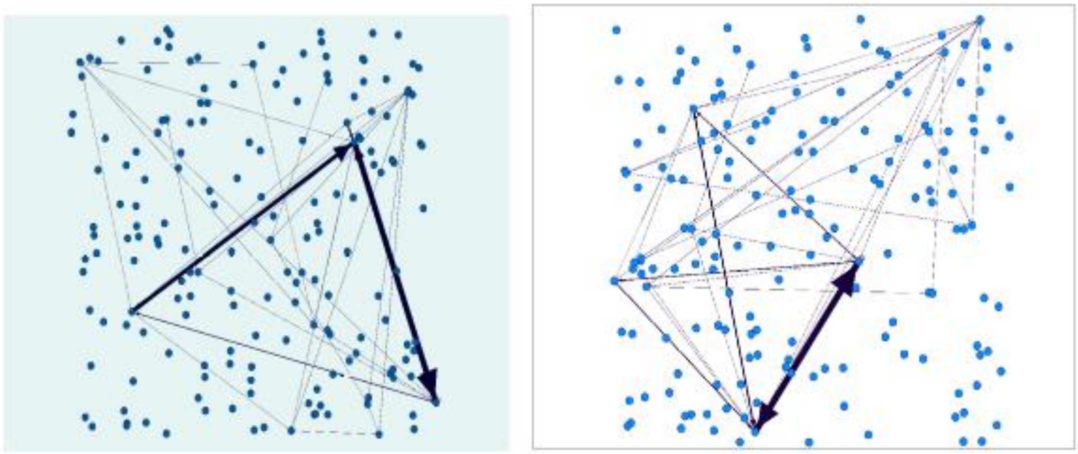
Traces of merged datasets from Materna workload trace (GWA-t-13_Materna-workload-traces) [12] and Fast storage trace from Bitbrain workload trace (GWA-t-12 Bitbrain workload trace) [13] are used in the Inter-cloud environment in order to generate new random workloads based on the set up of the environment.

5.3 Implementation Detail

For the implementation, a federation of clouds that resembles the Inter-cloud environment is used. The implemented Inter-cloud environment runs a service and gives output data that is used as a sample data for this work. Thus, for the purpose of implementing the Inter-cloud environment, the Materna workload trace, and Bitbrain workload trace are gathered from a variety of services. The output generated after running the traces in the Inter-cloud environment will be used as sample data for the implementation of the predictor model. The sample data will show the service behaviors in terms of resource workload issued from a different number of Cloud Service Providers. This data shows the behavior of the service from a resource demand point of view in the Inter-cloud environment. The implementation also includes the components of the Inter-cloud environment obeying a Service Level Agreement (SLA). The implementation of the Inter-cloud environment provides us a service behavior in terms of resource workload data that is used to train, test, validate, and run the neural network predictor. The implementation of the neural network starts after the output data from the Inter-cloud environment is preprocessed. Then the preprocessed data will be used to train, test, and validate the prediction model. The implementation of the proposed system follows the architecture developed and described in Chapter 4. The components and how they are implemented are described in detail as follows.

1) Inter-cloud Environment

As depicted in Figure 4.1 the implementation of the Inter-cloud environment as federation of CSP and Cloud Broker components is developed. The implementation of the environment uses four and five CSPs to gather the proper information of the service behavior in terms of resource workload. The other feature of the Inter-cloud environment is the migration of VM in a federation of clouds along with the service request. While implementing the environment migration of VM to a partner cloud in the federation is possible by rewrapping the VM in a service request. That means the current state of the VM is added to the service request and resumed to a new location. As depicted in Figure 5.1 (a) and (b) the service request migrated a VM from one CSP to another CSP within the Inter-cloud environment. The figure shows that the implementation of the Inter-cloud environment has transfer workloads by migrating VM from one CSP to another provider. The first Gephi graph illustration of VM migration within a federation of clouds, shows the migration of VMs between 4 Cloud Service Providers and the second graph shows the VM migration between 5 Cloud Service Providers. The figure depicted by using a directed graph to only show the process that the CSPs are undergoing while migrating the VMs with the respective service request in the environment. The nodes from the Gephi illustration resemble the hosts of the participating CSP and the edges resemble the migration of VMs from one CSP to another.



(a) VM migration between 4CSPs (b) VM migration between 5CSPs

Figure 5.1: *A sample Gephi illustration of VM migration within Inter-cloud environment*

2) The Prediction Component

The prediction component is responsible for the prediction of service behaviors in terms of resource workload based on the data generated from the Inter-cloud environment. Before jumping into building and training the neural network we have to first preprocess the dataset that is used to train and test the neural network. The ANN is trained by using the dataset that is gathered from the Inter-cloud environment. In this stage of the work irrelevant attributes that do not contribute to the prediction will be discarded. Data splitting and data transformation also takes place to prepare the dataset for the model-building task. As shown in Appendix-1 the Standard Scaler and Adam Optimizer in Keras library is used for data splitting and data transformation respectively. The predictor component responsible for forecasting using MLP with backpropagation training algorithm is implemented in Keras library [61].

5.4 Experiments and Experimental Results

To evaluate the performance of the proposed prediction model, experiments are performed and the experimental results are discussed in this Section. A neural predictor was developed with the aim of evaluating the performance of the neural network in predicting CPU utilization of Inter-cloud services. Even though there are many neural network applications freely and commercially available, for this study a new program is customized and used to extend various features and parameters using Keras library in Jupyter notebook. Moreover, new program development is the best way to gain a deep understanding of its internal operation. By using a wide range of inputs the networks are tested. The outcome showed that hundreds of inputs would result in longer training.

However, a smaller number of inputs will be insufficient to produce good results. Therefore, the network architectures that were tested have two hidden layers with 20, 10 hidden nodes and 30, 10 hidden nodes. In the experiments, these networks are evaluated with learning rate values at 0.01 and 0.0001 which are used to determine the rate at which the weights of the network are updated.

5.5 Experimentation setup

The multilayer perceptron feed-forward neural network methods will be employed to conduct the experiment. After the network is implemented, R^2 and MSE is used to evaluate the performance of the network. Network architecture with a large R^2 value on the testing data and small MSE will be used to build the model. To improve performance, the models were trained repeatedly until the best performance was obtained.

Based on the research objective of this study four experiments are designed. Experiment 1 and Experiment 2 are developed with the objective to analyze the services CPU utilization from the generated Materna workload data traces from 4CSPs and 5CSPs. Experiment 3 and Experiment 4 are developed with the objective to analyze the services CPU utilization from the generated Bitbrain workload data traces from 4CSPs and 5CSPs. We used four experiments to model the behaviour of services from two different resource workload data that are generated from 4CSPs and 5CSPs

On each experiments a network model of two hidden layer, one layer with 20 or 30 nodes and the other with 10 hidden nodes take place. To determine the effect of choice learning rate values on prediction accuracy, we used two learning rate values 0.01 and 0.0001.

5.6 Result and Discussion

In this Section, analysis and discussion of experimental results are presented.

Experiment 1: CPU utilization prediction of services from generated data of Materna workload traces from 4CSPs.

The sample data of the Materna workload traces from 4CSPs is presented in Table 4.1 in which columns from 5 to 12 are input neurons and column 13 is the output neuron. For the prediction, the sample data is transformed by the Min-Max normalization approach. When the experimentation uses two hidden layer with 20 and 10 nodes with learning rate of 0.01 the R^2 value will be 0.996 and MSE value of 1.987E-06. The experiment with Hidden layer of 30 and 10 node with the learning rate of 0.0001 yields the values of R^2 of 0.9875 and MSE of 1.39786E-05. When both of the network architecture uses the learning rate of 0.0001 the resulted value of R^2 is very large and MSE approximates its value to zero in both cases.

As depicted in Figure 5.2 the network architecture with the hidden neurons 20 and 10 have smaller MSE values than the network architecture that has hidden neurons of 30 and 10. Figure 5.3 shows the network architecture with the hidden neurons of 20 and 10 has R^2 values approximate to 1.00. The results from Experiment 1 shows that network model of 20 and 10 hidden layer nodes performs with higher accuracy than the network model of 30 and 10 hidden layer nodes. The scatter graph depicted in Figure 5.4 (a) and (b) shows that network model of 20 and 10 hidden layer nodes generalizes the target values better than network model with 30 and 10 hidden layer nodes.

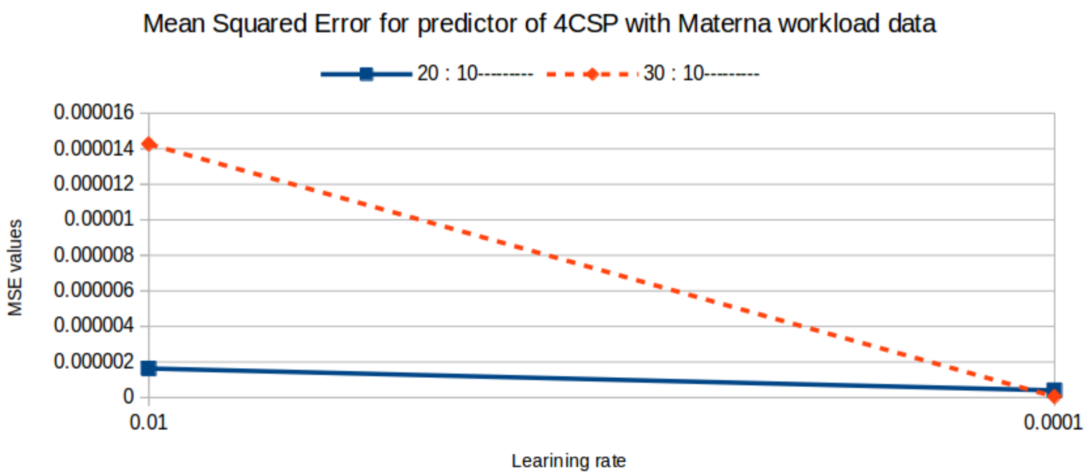


Figure 5.2: *Mean Squared Error for predictor of 4CSPs with Maternal workload data*

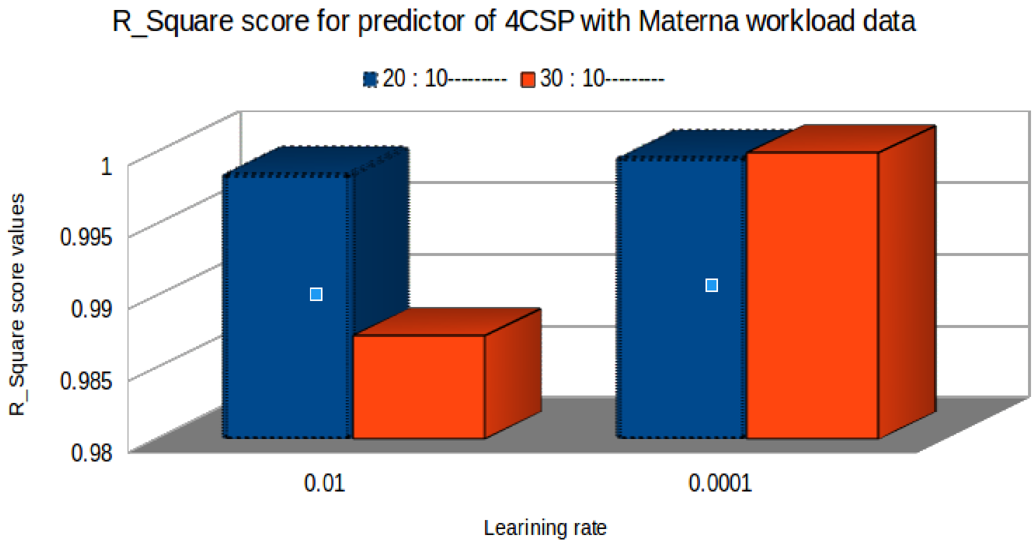
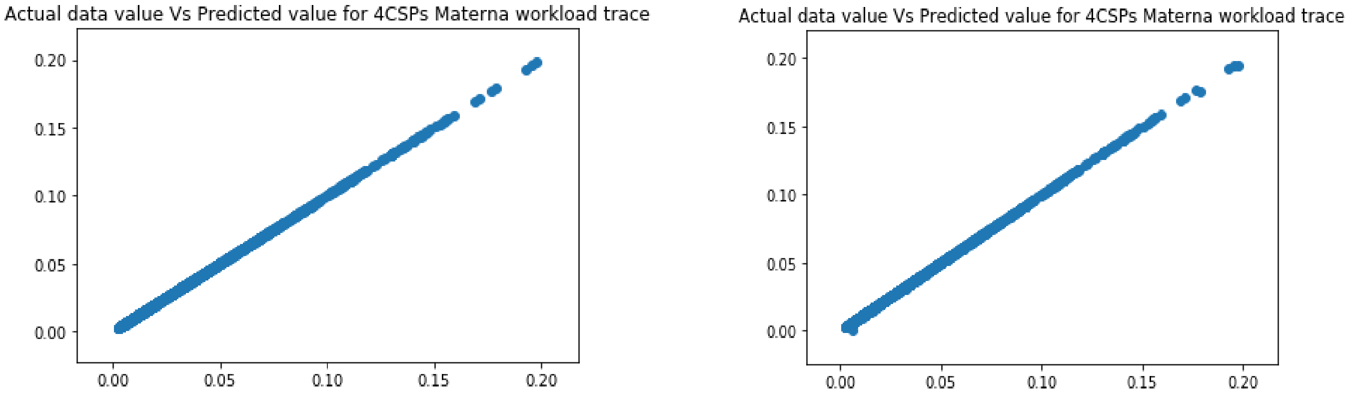


Figure 5.3: *R^2 values for predictor of 4CSPs with Maternal workload data*



(a) *Actual Vs Predicted values from model with 20 and 10 hidden nodes*

(b) *Actual Vs Predicted values from model with 30 and 10 hidden nodes*

Figure 5.4: *Scatter graph of Actual Vs Predicted values from Experiment 1*

Experiment 2: CPU utilization prediction of Inter-cloud services from the generated data of Materna workload traces from 5CSPs.

The sample data of the Materna workload traces from 5CSPs is presented in Table 4.2 in which columns from 5 to 12 are input neurons and column 13 is the output neurons. The experimentation of the model takes place with the same procedure as Experiment 1, the same network structures, and the learning rate values are applied. The result we found from the Experiment 2 is depicted in Figures 5.5 and 5.6. From the depicted figures the charts show that learning rate values applied on network model with 20 and 10 hidden layer nodes have R^2 values of 0.93786 and 0.98. The MSE values for the implementation of the 20 and 10 hidden layer network models with the two learning are 0.000215 and 0.00. By seeing the results from the charts, network model with 20 and 10 hidden nodes result in high prediction accuracy than network model with 30 and 10 hidden nodes. As seen in Experiment 1 the values of R^2 and MSE fluctuate when network structure with 30 and 10 hidden layer nodes are used for prediction. From the experimental results (Figure 5.5 and 5.6), it was observed that the model with 20 and 10 hidden layer nodes was able to generalize the output from a new future input data with less errors. The scatter graph depicted in Figure 5.7(a) and (b) shows the network model 20 and 10 hidden layer nodes generalize the target value out of the given sample data in a better way than the network model of 30 and 10 hidden layer node.

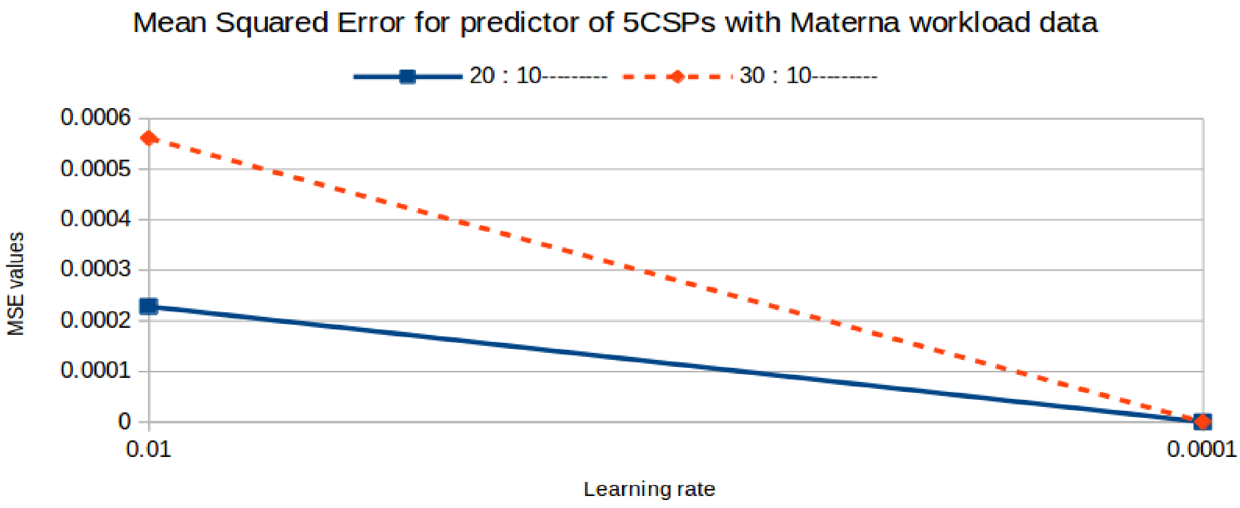


Figure 5.5: *Mean Squared Error for predictor of 5CSPs with Maternal workload data*

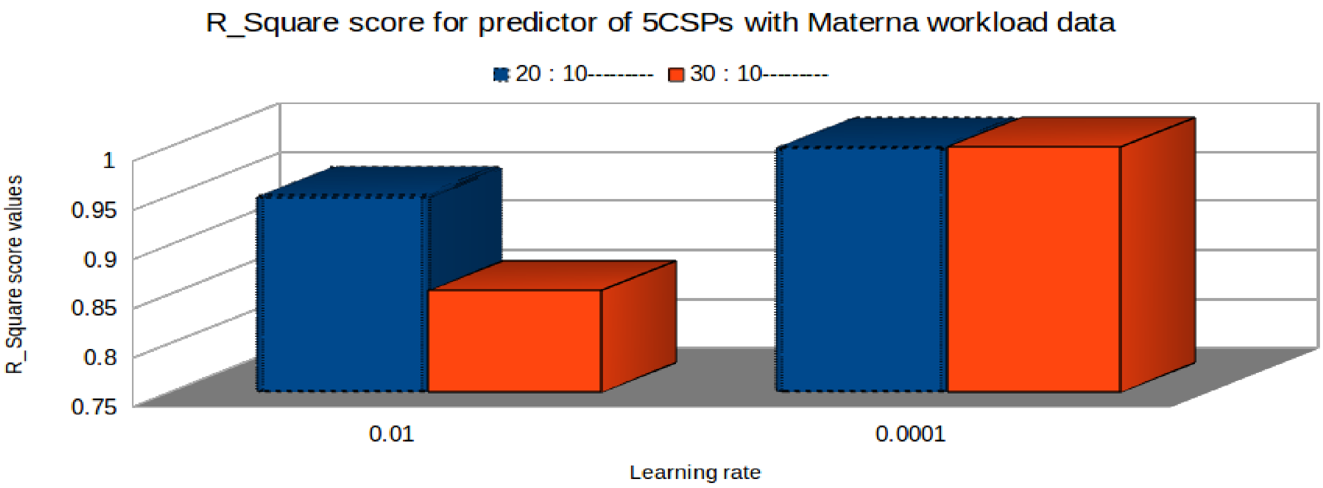
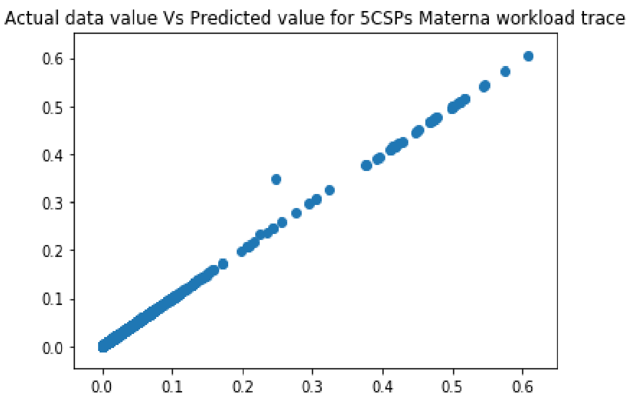
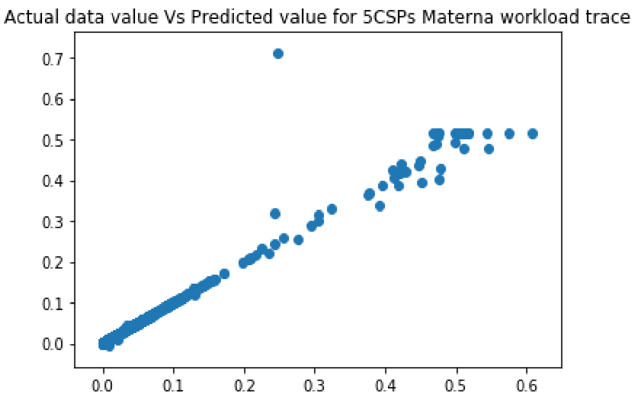


Figure 5.6: *R^2 values for predictor of 5CSPs with Maternal workload data*



a) *Actual Vs Predicted values from model with 20 and 10 hidden nodes*



(b) *Actual Vs Predicted values from model with 30 and 10 hidden nodes*

Figure 5.7: *Scatter graph of Actual Vs Predicted values from Experiment 2*

Experiment 3: CPU utilization prediction of services from generated data of Bitbrain workload traces from 4CSPs.

The sample data of the Bitbrain workload traces from 4CSPs presented in Table 4.3 will be used in Experiment 3. The results illustrated in Figure 5.8 and 5.9 shows that the network models with 30 and 10 hidden layer nodes yields higher R^2 values and MSE values of zero that the network models with 20 and 10 hidden layer nodes. The values of R^2 values and MSE fluctuate when 20 and 10 hidden layer network model is applied. Therefore, it is concluded that the models that has 30 and 10 hidden nodes model the target value with less error. This conclusion can also be seen with the scatter graphs illustrated in Figure 5.10 (a) and (b). As shown in the scatter graphs the predicted values are modeled better with the network with 30 and 10 hidden layer nodes.

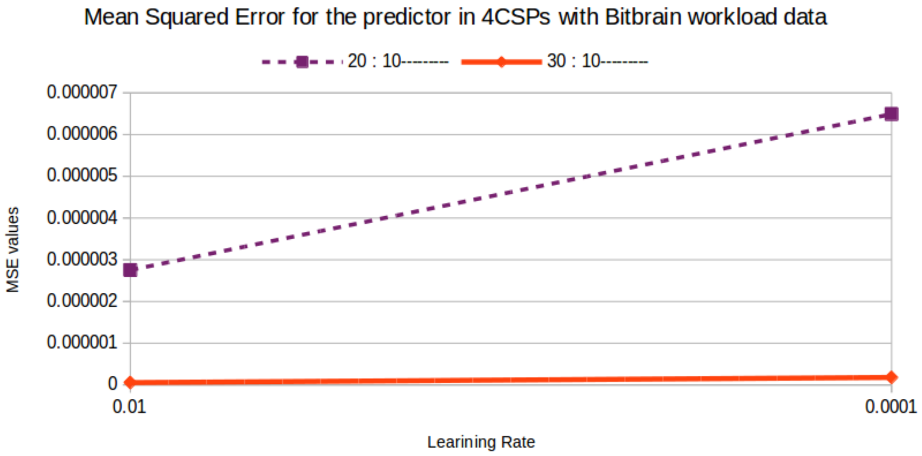


Figure 5.8: *Mean Squared Error for predictor of 4CSPs with Bitbrain workload data*

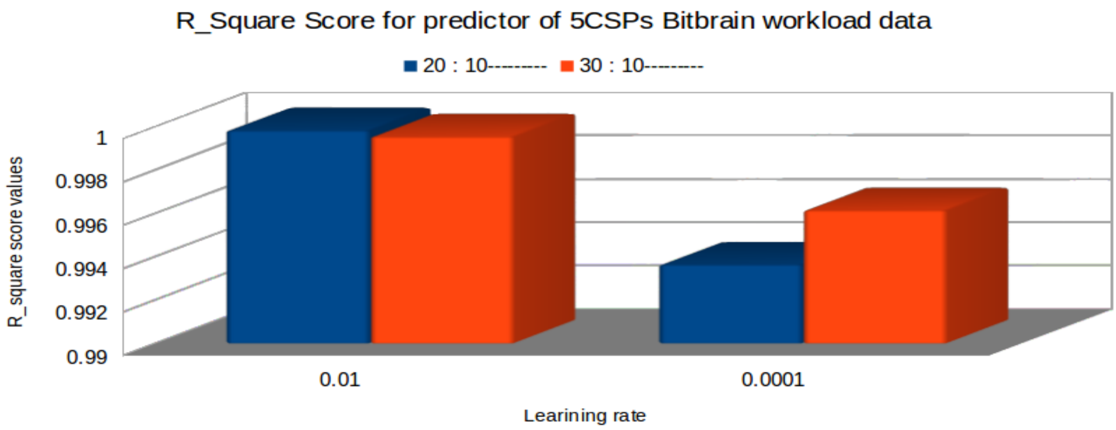
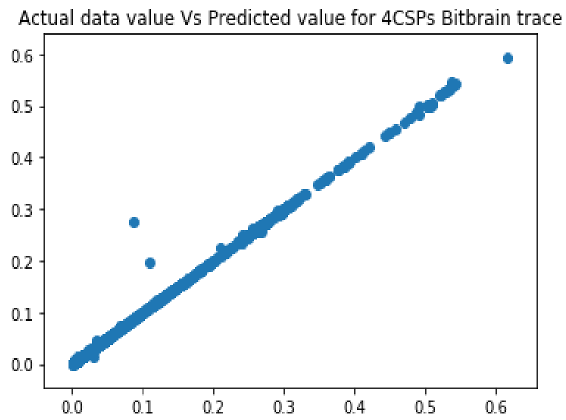
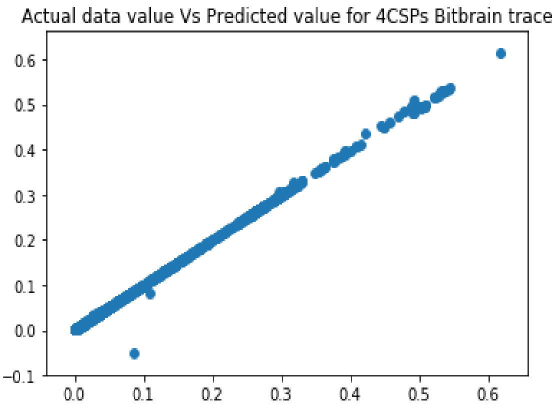


Figure 5.9: *R^2 values for predictor of 4CSPs with Bitbrain workload data*



(a) *Actual Vs Predicted values from model with 20 and 10 hidden nodes*



(b) *Actual Vs Predicted values from model with 30 and 10 hidden nodes*

Figure 5.10: *Scatter graph of Actual Vs Predicted values from Experiment 3*

Experiment 4: CPU utilization prediction of services from generated data of Bitbrain workload traces from 5CSPs.

In this experiment, we used the generated Bitbrain workload data from Table 4.4. As shown in Figure 5.11 and 5.12 the results from this experiment show that the R^2 value of the network model with 30 and 10 hidden layer nodes is higher than the model with 20 and 10 hidden nodes. The MSE values of the model with 30 and 10 hidden layer node are smaller. This shows that the neural model with a higher R^2 value and smaller MSE value which is the model with 30 and 10 hidden layer node will generalize the output out of sample data with less error. From the scatter graph illustration in Figure 5.13 (a) and (b) we can see that the network model of 30 and 10 hidden layer nodes performs better to find the target values than the model with 20 and 10 hidden nodes.

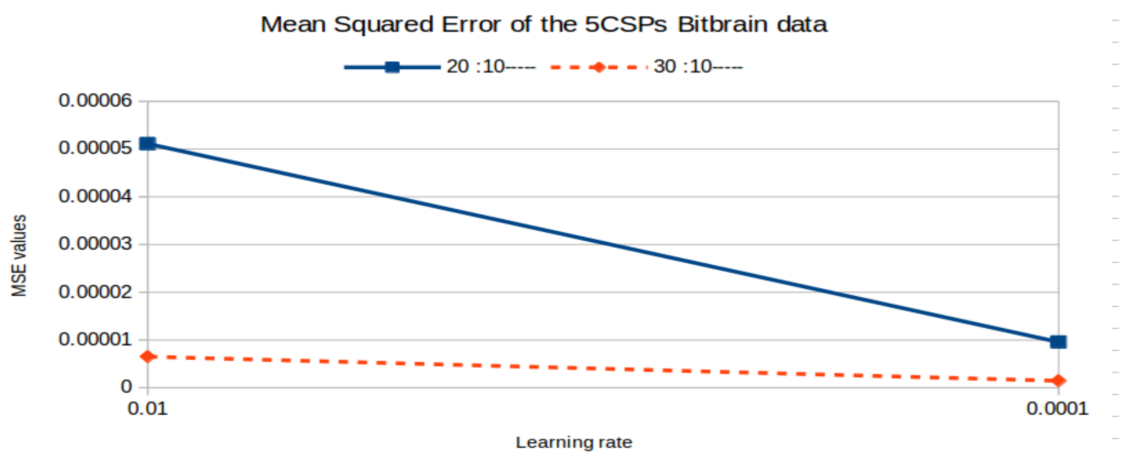


Figure 5.11: *Mean Squared Error for predictor of 5CSPs with Bitbrain workload data*

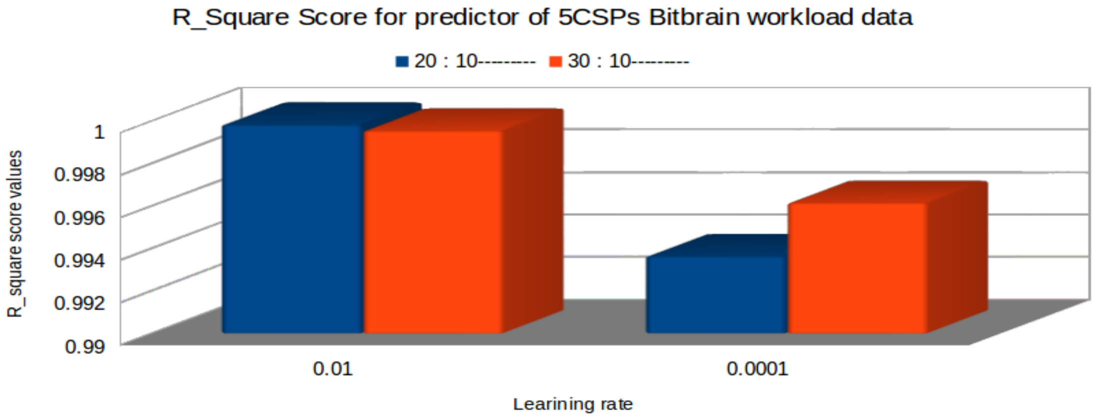
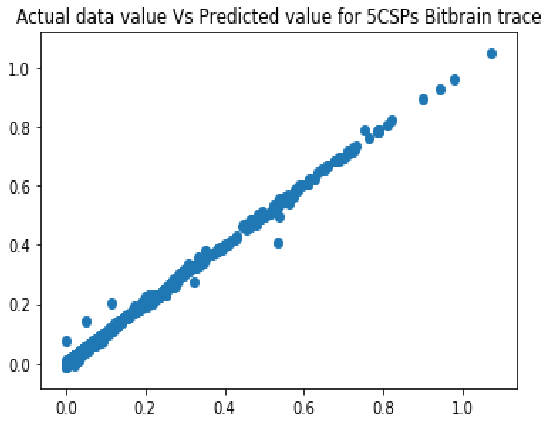
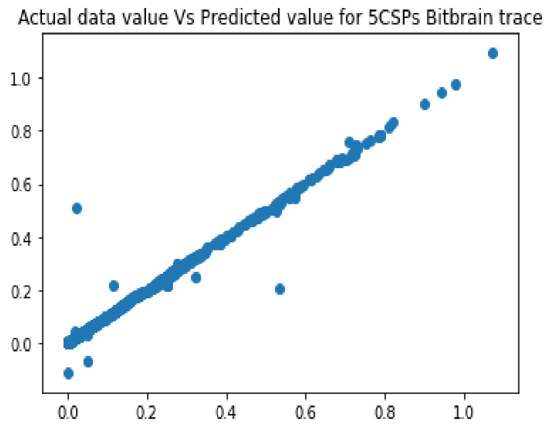


Figure 5.12: *R^2 values for predictor of 5CSPs with Bitbrain workload data*



(a) *Actual Vs Predicted values from model with 20 and 10 hidden nodes*



(b) *Actual Vs Predicted values from model with 30 and 10 hidden nodes*

Figure 5.13: *Scatter graph of Actual Vs Predicted values from Experiment 4*

5.7 Summary of Experimental Results

To evaluate the applicability and usability of the proposed system, the experimentations takes place and the resulting experimental results are the means for evaluating the system. From all the experimental results that are presented in the above charts, MSE has smaller values in Experiment 3 and 4 compared to Experiment 1 and 2 where the generated Bitbrain workload data is used. The low values of MSE terms imply that the neural network has successfully captured the hidden behavior of the resource workload and made highly accurate predictions. On the other hand, the R^2 values are larger in Experiment 1 and Experiment 3. We have observed that no combination of learning rate and network architecture is the best for all the workload data used, as values of MSE and R^2 keep changing with the changes in learning rate and network architecture, although the variation is quite small. From this, we can conclude the workload data can be modeled by choosing combination of learning rate and network architecture that yields lower MSE and large R^2 values.

From the experimentation, 0.0001 learning rate appears to generate the lowest values of MSE in Experiment 3 and 4 and the learning rate of 0.01 resulted in the lowest MSE values in Experiment 1 and 2, which means the most accurate results. In Experiments 1 and 2 the values of R^2 is larger if the learning rate value is 0.01 and R^2 is larger in Experiment 3 and 4 when 0.01 is the learning rate. However, on 3 out of the 4 Experiments, the network model of 30 and 10 hidden layer nodes with the learning rate of 0.0001 produce the best values of MSE and R^2. The only exception is Experiment 3, where the network model of 20 and 10 hidden layer nodes with the learning rate of 0.01 is used shows the best prediction performance. As a result, the learning rate of 0.0001 can be a good candidate for solutions aimed at high prediction accuracy.

Another significant factor that directly affects prediction accuracy is the shape of the workload data used to train, test and validate the network. Table 5.1 presents the numbers of workload data used in the four experiments. We present this table to get the full information of workload dataset numbers used and to find out the pattern of data distribution within the workload dataset. From this table testing set used on Experiment 4 is actually repeated as one part of the training set, though they are not exactly the same. The values of MSE on Experiments 3 and 4 is the smallest while on Experiments 1 and 2 MSE values became large.

Table 5.1: *Number of workload data used as training, validation and testing set in each experiment*

Name	Training set	Validation set	Testing set	Total
Data used in Experiment 1	7686	855	2136	10,677
Data used in Experiment 2	7210	802	2004	10,016
Data used in Experiment 3	7219	803	2006	10,028
Data used in Experiment 4	7216	802	2006	10,026

Table 5.2: *Mean and Variance of MSE values from each experiment*

Analysis using MSE value				
	Network model with 20 and 10 hidden nodes		Network model with 30 and 10 hidden nodes	
Experiments	Mean of MSE values	Variance of MSE values	Mean of MSE values	Variance of MSE values
Experiment 1	0.249	0.498	0.25	0.499
Experiment 2	0.25	0.499	0.249	0.496
Experiment 3	0.25	0.5	0.25	0.5
Experiment 4	0.247	0.494	0.25	0.499

Similar results are obtained with the Mean and variance of the prediction errors, as displayed in Table 5.2. The Mean of MSE in all experiments has values in the range of [0.24, 0.25]. Likewise, the variance values of MSE from all experiments range in [0.494, 0.5]. From the result, it's concluded that the shape of workload data used in the experimentations affect the variance of the errors and also may result in low prediction accuracy.

Chapter Six: Conclusion and Future Work

6.1. Conclusion

In this thesis, we investigated the problem of resource provisioning over the Inter-cloud environment. By only using the Inter-cloud application services behavior as a parameter for our work we developed a prediction model that will forecast the future surge of resources by the application services. The developed prediction model can be used as a device for other resource provisioning techniques, i.e, scaling over an Inter-cloud environment. The designed architecture for the proposed prediction model utilizing the measurements of application service behavior received from the Inter-cloud environment is presented.

First, the Inter-cloud environment is simulated using a simulator called FederatedClousim framework. The simulation incorporates all the components of the Inter-cloud environment. In this study, we have followed the same operational process of the FederatedCloudsim for the service requests. From the simulation of the Inter-cloud environment, data is generated that incorporates the behaviors of the application services in terms of resource workload. The gathered data from the Inter-cloud environment will go through a series of steps, i.e., attribute selection, data splitting, and data transformations in order to build a suitable prediction model. In this phase, the resource workload data generated from the Inter-cloud environment are normalized into the range [-1, 1]

After the suitable data for this study is prepared by undergoing a series of processes, the parameters and the structures need for a good prediction model is researched. Based on the findings of the suitable parameters and network structures the prediction model will be built. The application service behavior from the gathered data has a non-linear pattern. It is difficult to predict this data using traditional methods. To overcome this problem, machine learning models are used as an alternative to complex and non-linear forecast models. Machine learning models include artificial neural networks, support vector machines, and relevance vector machines.

From the comprehensive literature survey, an artificial neural network is selected for this study because it can model flexibly linear or non-linear relationships among variables. ANNs are capable to generalize to any desired accuracy. The most commonly used ANN prediction algorithm, the MLP neural network is to model the predictor. Finally, a three-layered MLP neural network with inputs neuron, one output neuron in the output layer, and two hidden layers and with reLu (Rectifier

Linear Unit) and sigmoid transfer functions were used. Then the next phase is model evaluation, the performance of the constructed models using different network structures was evaluated by R^2 and MSE. The prediction model with larger values of R^2 and smaller values of MSE is chosen. From the analysis of the results the network with 20 and 10 or 30 and 10 hidden layer nodes structures model all CPU utilization of the Inter-cloud services with varying degrees of accuracy.

Generally, CPU utilization generated from Materna workload data can be predicted by a network with 20 and 10 hidden layer nodes with higher accuracy while the CPU utilization generated from the Bitbrain workload can be predicted with acceptable error range which is less than 0.5. However the CPU utilization generated from Bitbrain workload data can be predicted by the models constructed with hidden layer nodes of 30 and 10 with higher accuracy while the CPU utilization generated from the Materna workload can be predicted by models constructed with hidden layer nodes of 20 and 10 with high accuracy.

6.2 Contribution

The overall study has the following contributions:

- Prediction model that can forecast the future behaviors of services hosted in the Inter-cloud environment.
- The proposed prediction model can be used to avoid resource provisioning issues, i.e., over-fitting and under-fitting of resources.

6.3 Future Work

After the predictor model is designed and evaluated the output results will be stored for further use. The future work of our study will be further development of CB component that will devise a strategies based on the output results for the predictor component to select the appropriate CSP.

References

[1] Chunye Gong, Jie Liu, Qiang Zhang, Haitao Chen, and Zhenghu Gong, "The Characteristics of Cloud Computing", Proceedings of the 39th International Conference on Parallel Processing Workshops (ICPPW), IEEE Computer Society 2010, pp.275-279

[2] NIST, "Cloud Computing Synopsis and Recommendations," http://csrc.nist.gov/ publications/ nistpubs/800-146/sp800-146.pdf, 2012, Accessed, October 2018.

[3] B. Kezia Rani B. Padmaja Rani A. Vinaya Babu, "Cloud Computing and Inter-Clouds – Types Topologies and Research Issues", Computer Science, Vol. 50, pp. 24-29, 2015.

[4] Forum GICTF, “Use cases and functional requirements for inter-cloud computing”, GICTF White Paper, 2010.

[5] Rajkumar Buyya, Rajiv Ranjan, and Rodrigo N. Calheiros, “InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Service”, Proceedings of the 10th international conference on Algorithms and Architecture for parallel processing, May 21-23, 2010, Busan, South Korea.

[6] Sadeka Islam, Jacky Keung, Kevin Lee, and Anna Liu ,“Empirical prediction models for adaptive resource provisioning in the cloud”, Future Generation Computer Systems 28, 2012.

[7] Zhenhuan Gong, Xiaohui Gu, and John Wilkes, PRESS: PRedictive Elastic ReSource Scaling for cloud systems, NSF CNS091586, CNS0915567, Google Research Award, and IBM Faculty Award, 2010.

[8] Andreas Kohne, Marc Spohr, Lars Nagel, and Olaf Spinczyk. FederatedCloudSim: SLA-aware Federated Cloud Simulation Framework. In Proceedings of the 2nd International Workshop on CrossCloud Systems, CCB , New York, USA, 2014. ACM.

[9] A. Sheth and A. Ranabahu, "Semantic Modeling for Cloud Computing, Part I & II," IEEE Internet Computing Magazine, Vol. 14, pp. 81-83, 2010.

[10] J. Bozman and G. Chen, "Cloud Computing: The Need for Portability and Interoperability," IDC Analyze the Future, Sponsored by Red Hat, Inc., 2010.

[11] Feitelson, D.G., “Workload Modelling for Computer Systems Performance Evaluation” Cambridge University Press, 2012.

[12] “Materna workload Traces”, http://atlarge.ewi.tudelft.nl/graphalytics/cloud/GWA-T-13_Materna-Workload-Traces.zip, Accessed September, 2020

[13] “Bitbrain fastStorage workload Traces”, http://gwa.ewi.tudelft.nl/fileadmin/pds/trace-archives/grid-workloads-archive/datasets/ gwa-t-12/fastStorage.zip. Accessed September,2020.

[14] John Harauz, Lorti M. Kaunan. and Bruce Potter, "Data Security in the World of Cloud Computing", IEEE Security & Privacy, Copublished by the IEEE Computer and Reliability Societies, July/August 2009.

[15] Marios D. Dikaiakos, George Pall, Dimitrios Katsaros, Pankaj Mehra, and Athena Vakali, "Cloud computing: Distributed Internet Computing for IT and Scientiic Research", IEEE Internet Computing, Published by the IEEE Computer Society, September/October 2009.

[16] Williams,“The economics of cloud computing: An overview for decision makers.”, Indianapolis, IN: Cisco Press, 2012

[17] Blair H. Sheppard, Hartwick J, and Warshaw, “The theory of reasoned action: A meta-analysis of past research with recommendations for modifications and future research”, Journal of Consumer Research, 15, 325-344.

[18] T.Swathi, K.Srikanth, and S. Raghunath Reddy, “Virtualization in Cloud Computing”, IJCSMC, Vol. 3, Issue. 5, May 2014, pp.540 – 546.

[19] Joel Gibson, Darren Eveleigh, Robin Rondeau, and Qing Tan, “Benefits and Challenges of Three Cloud Computing Service Models”, IEEE, 2012.

[20] Malhotra. L, Agarwal. D, and Jaiswal, “Virtualization in Cloud Computing”, Information Technology Software Engineering, 2014.

[21] Zhen Xiao, Weijia Song, and Qi Chen, “Dynamic Resource Allocation using Virtual Machines for Cloud Computing Environment”, IEEE Transactions on Parallel and Distributed Systems ,volume: 24, Issue 6: June 2013)

[22] Sijin He, Li Guo, Yike Guo, Chao Wu, Moustafa Ghanem, and Rui Han, “Elastic Application Container: A Lightweight Approach for Cloud Resource Provisioning”, 26th IEEE International Conference on Advanced Information Networking and Applications, 2012.

[23] R. Jeyarani, N. Nagaveni, and R. Vasanth Ram,” Design and implementation of adaptive power-aware virtual machine provisioner (APA-VMP) using swarm intelligence”, Future Generation Computer Systems 28, 2012, PP. 811–821.

[24] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, “Black-box and gray-box strategies for virtual machine migration,” in Proc. of the Symposium on Networked Systems Design and Implementation (NSDI’07), Apr. 2007.

[25] N. Bobroff, A. Kochut, and K. Beaty, “Dynamic Placement of Virtual Machines for Managing SLA Violations,” Proc. IFIP/IEEE Int’l Symp. Integrated Network Management (IM ’07), pp. 119-128, May 2007.

[26] Gaurav Banga, Peter Druschel, and Jeffrey C. Mogul, “Resource Conainers: A New Facility for Resource Management in Server Systems”, In Proceedings of the Symposium on Operating Systems Design and Implementation, 1999.

[27] “Use Cases and Functional Requirements for Inter-Cloud Computing”, Technical Report, Global Inter-Cloud Technology Forum, August 9, 2010.

[28] Ali Yadavar Nikravesh, Samuel A. Ajila, and Chung-Horng Lung, “Towards an Autonomic Auto-Scaling Prediction System for Cloud Resource Provisioning”, 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, 2015.

[29] Bhavani B .H, and H. S Guruprasad, “Resource Provisioning Techniques in Cloud Computing Environment: A Survey“, International Journal of Research in Computer and Communication Technology, Vol 3, Issue 3, March 2014.

[30] Mehdi Khashei, and Mehdi Bijari, “An artificial neural network model for timeseries forecasting”, Expert Systems with Applications 37,2010, pp. 479–489.

[31] Piyush Shivam, Shivnath Babu, and Jeffrey S. Chase, “Learning application models for Utility resource planning”, IBM and National Science Foundation, 2008.

[32] Yu L., Wang S, and Lai K., “A novel nonlinear ensemble forecasting model incorporating GLAR and ANN for foreign exchange rates”, Computers and Operations Research, 32, 2523–2541, 2005.

[33] G. Zhang, “Time series forecasting using a hybrid ARIMA and neural network model”, Neurocomputing 50, 159–175, http://dx.doi.org/10.1016/S0925-2312(01)00702-0, 2003.

[34] Akindele A. Bankole and Samuel A. Ajila, “Predicting Cloud Resource Provisioning using Machine Learning Techniques”, 2013 26th IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), 2013.

[35] G. Box and G. Jenkins, “Time Series Analysis: Forecasting and Control”, Holden-Day, 1976.

[36] Truong Vinh, Truong Duy, Yukinori Sato, and Yasushi Inoguchi,” Improving Accuracy of Host Load Predictions on Computational Grids by Artificial Neural Networks”, 20 February 2010.

[37] G. Zhang, B.E. Patuwo, and M.Y. Hu, “Forecasting with artificial neural networks: The State of the Art”, International Journal of Forecasting, 14-1, pp. 35-62, 1998.

[38] Mateusz Guzek, Alicja Gniewek, Pascal Bouvry, Jedrzej Musial, and Jacek BlazewiczJedrzej Musial and Jacek Blazewicz ., “Cloud Brokering: Current Practices and Upcoming Challenges.” IEEE Cloud Computing, March 2015.

[39] Hosseini, H., Luo, D., and Reynolds, K. J.,“The comparison of different feed forward neural network architectures for ECG signal diagnosis”, Medical Engineering and Physics, 2006.

[40] Zhang, Guoqiang, B. Eddy Patuwo, and Michael Y Hu., "Forecasting with artificial neural networks:: The state of the art", International journal of forecasting 14, Vol 1: 35-62, 1998.

[41] Lendasse, Amaury, John Lee, Eric De Bodt, Vincent Wertz, and Michel Verleysen. "Approximation by radial basis function networks", pp. 203-214. Springer US, 2003.

[42] L.A. Zadeh, “The concept of a linguistic variable and its application to approximate reasoning”, International Information Science 8 ,1975, pp. 199–249.

[43] Q. Song, and B.S. Chissom, “Forecasting enrollments with fuzzy time series—(part I), Fuzzy Sets and Systems” 1993.

[44] Q. Song, and B.S. Chissom, "Forecasting enrollments with fuzzy time series—(part II), Fuzzy Sets and Systems", 1994.

[45] Taskaya, T., and Casey, M. C.," A comparative study of autoregressive neural network hybrids. Neural Networks", pp. 781–789, 2005.

[46] Baxt, W. G., "Improving the accuracy of an artificial neural network using multiple differently trained networks", Neural Computation 4, pp. 772–780, 1992.

[47] Peter Zhang,"A neural network ensemble method with jittered training data for time series forecasting", Information Sciences, 177, pp. 5329–5346, 2007.

[48] Khashei, M., Hejazi, S. R., and Bijari, M., "A new hybrid artificial neural networks and fuzzy regression model for time series forecasting" , Fuzzy Sets and Systems, 159, pp. 769–786, 2008.

[49] Evgueni D, Rodrigo F, "A Model for Automatic On-Line Process Behavior Extraction, Classification and Prediction in Heterogeneous Distributed Systems", Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid'07), 2007.

[50] Evgueni D, Rodrigo F, "A novel approach for distributed application scheduling based on prediction of communication event", Future generation computer systems 26,pp. 740-752, 2010.

[51] V. Nelson, and V. Uma, "Semantic based Resource Provisioning and Scheduling in Inter-cloud Environment", IEEE, 2012.

[52] HyunjeongYoo, CinyoungHur, Seoyoung Kim, and YoonheeKim, "An ontology-based resource selection service on science cloud."International Journal of Grid and Distributed Computing Vol. 2, No. 4, December, 2009.

[53] VMware: Migrate Virtual Machines with Zero Downtime, http://www.vmware.com/, Accessed September 2019.

[54] Nour El, Houda Bouzerzour, Souad Ghazouani, and Yahya Slimani, "A survey on the service interoperability in cloud computing: Client-centric and provider-centric perspectives", December, 2019.

[55] Barham et al., “Xen and the Art of Virtualization.”, In: Proceedings of the the 19th ACM Symposium on Operating Systems Principles. ACM Press, New York, 2003.

[56] Susanta Nanda, and Tzi-cker Chiueh, “A Survey on Virtualization Technologies”, January 2005

[57] Rabi Prasad Padhy, Manas Ranjan Patra, and Suresh Chandra Satapathy, “virtualization techniques & technologies: state-of-the-art“, International Journal of Computer Networks And Applications (IJCNA), 2015.

[58] Mateusz Guzek, Alicja Gniewek, Pascal Bouvry, Jedrzej Musial and Jacek BlazewiczJedrzej Musial and Jacek Blazewicz ., “Cloud Brokering: Current Practices and Upcoming Challenges.” IEEE Cloud Computing, March 2015.

[59] Ferrer AJ, Hernández F, Tordsson J, Elmroth E, Ali-Eldin A, Zsigri C, Sirvent R, Guitart J, Badia RM, Djemame K, Ziegler W, Dimitrakos T, Nair SK, Kousiouris G, Konstanteli K, Varvarigou T, Hudzia B, Kipp A, Wesner S, Corrales M, Forgó N, Sharif T, Sheridan C. OPTIMIS: “a holistic approach to cloud service provisioning”, Future Generation Computer Systems 2012; pp:66–77.

[60] Lucas Simarro J, Moreno-Vozmediano R, Montero R, and Llorente I, “Dynamic placement of virtual machines for cost optimization in multi-cloud environments”, In Proceedings of the International Conference on High Performance Computing and Simulation (HPCS 2011). IEEE: Istanbul, pp:1–7, Turkey, 2011.

[61] Keras Library, https://keras.io/, Accessed July 2020.

[62] FederatedCloudsim Simulator, https://ess.cs.tu-dortmund.de/Software/FederatedCloudSim/, Accessed July 2020.

[63] Jupyter notebook, https://jupyter.org/, Accessed July,2020.

[64] Rajkumar Buyya, Rajiv Ranjan and Rodrigo N. Calheiros1, "Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities", Proceedings of the 7th High Performance Computing and Simulation (HPCS 2009) ,July 2009

Appendix-1: Sample Code Segment for Prediction Component

#Part 1 - Data Preprocessing

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

#Importing the dataset
dataset = pd.read_csv('/home/abuha/Desktop/NN/4CSPsample.csv')
X = dataset.iloc[:, 5:13].values
y = dataset.iloc[:,12].values

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

import numpy as np
training_set_shape = X_train.shape
print(training_set_shape)
training_set_shape = X_test.shape
print(training_set_shape)
training_set_shape = y_train.shape
print(training_set_shape)
all_set_shape = X.shape
print(all_set_shape)
```

 # Part 2 - Now let's develop the ANN!

```
# Importing the Keras libraries and packages
import tensorflow as tf
from tensorflow import keras
from tensorflow.python.keras.models import Sequential #used to initialize the NN
from tensorflow.python.keras.layers import Dense  #used to build the hidden Layers
from tensorflow.python.keras.layers import Dropout
from tensorflow.python.keras.optimizers import Adam
from tensorflow.python.keras import backend
from tensorflow.python.keras import backend as K
from sklearn.metrics import r2_score

def rmse(y_true, y_pred):
        return backend.sqrt(backend.mean(backend.square(y_pred - y_true), axis=-1))
```

```
def r2_keras(y_true, y_pred):
    SS_res =  K.sum(K.square(y_true - y_pred))
    SS_tot = K.sum(K.square(y_true - K.mean(y_true)))
    return ( 1 - SS_res/(SS_tot + K.epsilon()))
# Initialising the ANN
classifier = Sequential()

# Adding the input layer and the first hidden layer with dropout
classifier.add(Dense(units= 30, activation='relu', kernel_initializer='glorot_uniform',
input_dim=8))

# Adding the second hidden layer
classifier.add(Dense(units = 10, kernel_initializer = 'uniform', activation = 'relu'))

# Adding the output layer
classifier.add(Dense(units = 1, kernel_initializer = 'uniform', activation = 'linear'))

 classifier.summary()
```

```
# output
Layer (type)              Output Shape            Param #
=================================================================
dense_1 (Dense)           (None, 30)              270
_________________________________________________________________
dense_2 (Dense)           (None, 10)              310
_________________________________________________________________
dense_3 (Dense)           (None, 1)               11
=================================================================
Total params: 591
Trainable params: 591
Non-trainable params: 0
_________________________________________________________________
```

```
# Compiling the ANN
classifier.compile(optimizer = Adam(lr=0.001), loss = 'mse', metrics = ['mse', rmse])

# Fitting the ANN to the Training set
history=classifier.fit(X_train, y_train, validation_split=0.1, batch_size = 10, epochs = 100 ,
verbose= 2)
```

```
# output
Train on 13071 samples, validate on 1453 samples
Epoch 1/100
 - 2s - loss: 13.4160 - mean_squared_error: 13.4160 - rmse: 1.2301 - val_loss: 0.3366 - val_mean_squared_error:
0.3366 - val_rmse: 0.3247
Epoch 2/100
 - 2s - loss: 0.1307 - mean_squared_error: 0.1307 - rmse: 0.2283 - val_loss: 0.0493 - val_mean_squared_error:
0.0493 - val_rmse: 0.1434
Epoch 3/100
 - 2s - loss: 0.0252 - mean_squared_error: 0.0252 - rmse: 0.0933 - val_loss: 0.0142 - val_mean_squared_error:
0.0142 - val_rmse: 0.0723
.
.
.
```

```
Epoch 100/100
- 2s - loss: 3.5592e-07 - mean_squared_error: 3.5592e-07 - rmse: 1.9482e-04 - val_loss: 4.6100e-08 -
val_mean_squared_error: 4.6100e-08 – val_rmse: 4.3560e-05
```

```
y_pred=classifier.predict(X_test)

#for i in y_pred:
#   print predicted values
for i in range(len(X_test)):
    print((y_pred[i]))

#actual test data
for i in range(len(X_test)):
    print((y_test[i]))

score = classifier.evaluate(X_test,y_test, verbose=0)
trainscores = classifier.evaluate(X_train,y_train, verbose=0)
r2 = r2_score(y_test, y_pred)

print('Testing Data MSE value:', score[0])
print('Training Data MSE value:', trainscores[0])

print('R2_score value:', r2)

#print('Average of MSE value for training data', trainscores[0] / y_train.shape)

from statistics import *
print('MSE Mean: %.3f, MSE Standard Deviation: %.3f' % (mean(score), stdev(score)))
print('MSE Mean: %.3f, MSE Standard Deviation: %.3f' % (mean(trainscores), stdev(trainscores)))

plt.plot(y_test[i], label= 'actual test data')

plt.plot(y_pred[:,0], label= 'predicted values')
plt.legend()
plt.show()

# save model and architecture to single file
import os
if os.path.isfile("Seven4model.h5") is False:
    classifier.save("Seven4model.h5")
print("Saved model to disk")
```

Declaration

I, the undersigned, declare that this thesis is my original work and has not been presented for a degree in any other university, and that all source of materials used for the thesis have been duly acknowledged.

Declared by:

Name: ____

Signature: ____________________________________

Date: ____

Confirmed by advisor:

Name: ____

Signature: ____________________________________

Date: ____

Place and date of submission: Addis Ababa, March 2021.