WORKLOAD OPTIMIZATION IN FEDERATED CLOUD ENVIRONMENT

Thesis

Submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

by

SHISHIRA S R

DEPARTMENT OF MATHEMATICAL & COMPUTATIONAL SCIENCES

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALORE - 575025

January, 2022

Dedicated to
My Teachers & Family

DECLARATION

By the Ph.D. Research Scholar

I hereby *declare* that the Research Thesis entitled **WORKLOAD OPTIMIZATION IN FEDERATED CLOUD ENVIRONMENT** which is being submitted to the **National Institute of Technology Karnataka, Surathkal** in partial fulfillment of the requirements for the award of the Degree of **Doctor of Philosophy** in **Department of Mathematical and Computational Sciences** is a *bonafide report of the research work carried out by me.* The material contained in this Research Thesis has not been submitted to any University or Institution for the award of any degree.

Shishir

Place: NITK, Surathkal.

Date: 07-01-2022

(Shishira S R)

155048MA15F07

Department of Mathematical and Computational Sciences

CERTIFICATE

This is to *certify* that the Research Thesis entitled **WORKLOAD OPTIMIZATION IN FEDERATED CLOUD ENVIRONMENT** submitted by **Shishira S R**, (Reg. No.: 155048MA15F07) as the record of the research work carried out by her, is *accepted as the Research Thesis submission* in partial fulfillment of the requirements for the award of degree of **Doctor of Philosophy**.

AKsamy

Prof. A. Kandasamy

Research Supervisor

RMadhusudhan 7/1/22

Chairman - DRPC

(Signature with Date and Seal)

Chairman
DUGC / DPGC / DRPC
Dept. of Mathematical and Computational Sciences
National Institute of Technology Karnataka, Surathkal
MANGALORE - 575 025

ACKNOWLEDGMENT

I would like to thank all those people who have made this research work possible.

First and foremost, I would like to express my sincere gratitude towards my research supervisor *Prof. A. Kandasamy*, NITK Surathkal, for his enormous support, consistent encouragement and continuous guidance throughout my research. His advice on both research and on my career have been priceless.

I would also like to thank the Research Progress Assessment Committee members: *Prof. B R Shankar*, Dept. of MACS and *Prof. Lakshman Nandagiri*, Dept. of Water Resources and Ocean Engineering for their invaluable feedback and useful comments.

I am thankful to the former HOD of Dept. of MACS, *Prof. Santhosh George* and *Prof. B R Shankar* and the present HOD, *Prof. Shyam S. Kamath*, for providing better facilities and good working environment for research.

I would like to thank all the faculty members of Dept. of MACS and special thanks to *Prof. Chandrasekaran K*, Dept. of CSE for his inspiration and valuable guidance in my research. I also thank all the non-teaching staff for their support and help. I am grateful to my fellow research scholars for their support in my research life. Thanks to the National Institute of Technology Karnataka for making my Ph.D. study possible with its prestigious scholarship.

Finally, I extend my heartfelt thanks to my family for their moral support during the course of my research work.

Place: Mangalore Shishira S R

Date: 07-01-2022

ABSTRACT

Cloud computing is an essential paradigm for processing, computing, storing, and communication bandwidth. It offers services on an on-demand basis for the user, that is, pay per use. Cloud computing consists of numerous resources, including the provision of networks, databases for storage, servers, virtual machines, and potential application. It is a widely used technique to handle large amounts of data as it provides versatility and functionality for optimization. Customers submit their request for data exchange and to store it in an existing cloud environment. The customer has a huge advantage in paying for the currently required services. In a federated cloud environment, one or more cloud service providers share their servers to handle user requests. It promotes cost savings, service utilization, and performance enhancement. Clients would benefit as a Service Quality Agreement exists between the two. The Cloud federation is an evolving technology through which cloud service providers cooperate to provide clients with customized services to enjoy the real benefits of Cloud Computing. The federated service provider achieve better resource usage and Quality of Service by cooperation, thereby enhancing their market prospects.

Workloads are the collection of raw inputs provided to the processing arhcitecture. Based on the successful processing of workloads, efficiency can be assessed. Different workloads have distinct feature sets. The secret to making optimal configuration decisions and improving system performance is by recognizing the characteristics of workloads. Multiple requests are handled quickly under the dynamic cloud environment, which contributes to the resource allocation problem.The cloud will maintain the workflow active through the proper allocation of resources, virtualization software, or repositories. However, the precise load estimation model is important for efficient management of resources.

It is hard to manage a large number of workloads in an enterprise cloud system. Workloads are the sum of data for processing that are provided to the hardware resource. Its behavior and characteristics play an important role in the efficient processing of resource requests. It is also difficult to predict the existence of workloads if they alter excessively. In this thesis, we propose a conceptual framework for efficient prediction and optimization of workloads that can be easily adapted to a system to address this problem to address this problem. Serving the request in considerably less time leads to an issue with resource allocation. In order to auto-scale the resources, it is more comfortable to have previous awareness of the incoming loads. For the better prediction of workloads in the cloud world, a novel architecture is proposed. Predicted workloads could also be configured smoothly for better use without waving off, the SLA negotiated between the provider and customers. Three essentials for the management of cloud resources are considered in the proposed Fitness Function Extraction Model, i.e. CPU, Disk, and Memory storage.

This thesis proposes a BeeM-NN architecture by incorporating Workload Neural Network Algorithm and Novel Bee Mutation Optimization Algorithm into a cloud environment for optimized workload prediction. The proposed model initially includes the Fitness Function Extraction Algorithm to retrieve the attribute samples from the Microsoft Azure traces. With the Novel Bee Mutation Optimization Algorithm in the cloud, the expected QoS are optimized. The developed model is tested using the federated cloud service providers workload data traces and is analyzed with the benchmark methods. The result indicated that the proposed model obtained higher accuracy than the existing systems with optimum efficiency in resource and cost usage.

Keywords: Cloud Workloads, Workload chracterization, Prediction, Optimization, Artificial Neural Networks, Machine Learning, Feature extraction, Meta-heuristic Techniques, Genetic Algorithm, Artificial Bee Colony.

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

ABC	: Artificial Bee Colony
ACO	: Ant Colony Optimization
ANN	: Artificial Neural Network
BLSTM	: Bidirectional Long Short-Term Memory
CPU	: Central processing unit
DBN	: Deep Belief Network
FFEA	: Fitness Feature Extraction Algorithm
GA	: Genetic Algorithm
HTTP	: Hypertext Transfer Protocol
I/O	: Input-Output
LAN	: Local area network
LR	: Linear Regression
LSTM	: Long Short-Term Memory
NBMOA	: Novel Bee Mutation Optimization Algorithm
PM	: Prediction model
PSO	: Particle Swarm Optimization
QoE	: Quality of Experience
QoS	: Quality of Service
ReLu	: Rectified Linear unit
RNN	: Recurrent Neural Network
SLA	: Service Lebel Agreement
VM	: Virtual Machine
WNNA	: Workload Neural Network Algorithm

Chapter 1

INTRODUCTION

Cloud computing is a model that serves the request of the consumer, which gives users on-demand access to services. Cloud computing comprises various resources, such as network distribution, databases for storage, servers, virtual machines, and potential application (Buyya et al., 2010). In computing over the Internet, cloud computing plays an important role, where the user does not have to use extensive infrastructure or expensive applications. It is possible to use the program on a rental basis and only pay for the purchase or demand services (Bohn et al., 2011). Cloud services are in huge demand since many users host their cloud apps on various platforms. The essential components of the cloud environment can be seen in Figure 1.1. There are a large number of services available from cloud service providers to support the user. There is a tremendous data loss for users in a single deployment cloud when there is a natural catastrophe or other security threats on the network (Badger et al., 2012). Hence by sharing the infrastructure and serving the demand, multiple cloud deployment overcomes this disadvantage.

The cloud service broker functions between the provider and the customer as an intermediate. The broker helps to transfer the customers request to the provider and get the best service provider to serve the customers request (Tordsson et al., 2012). Regardless of the geographically located servers, a broker helps to track, maintain, and access the cloud services from different providers. The SLA for accessing the service is

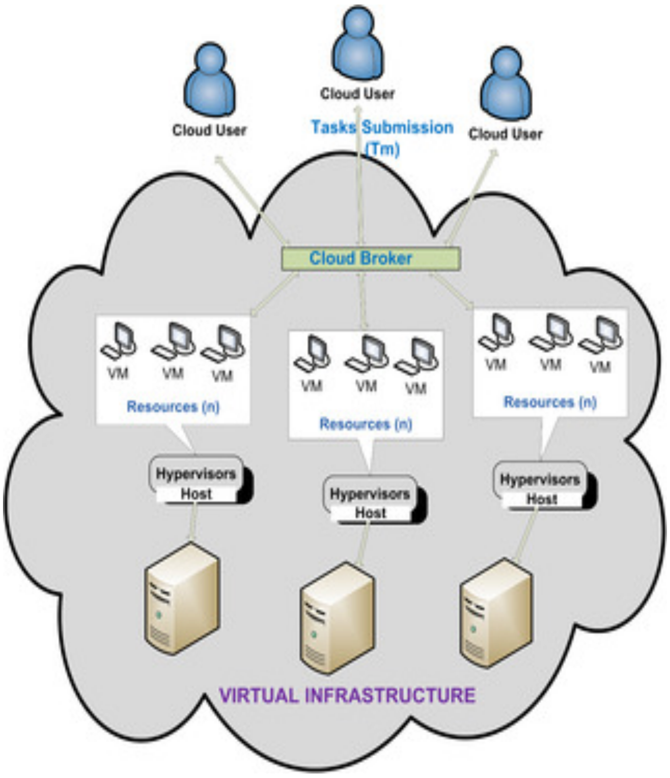


Figure 1.1 Key components of cloud environment.

signed after the broker has selected the best cloud service provider to support a customer order. Hence, compared to a single deployment cloud, the quality of service is high in this deployment platform. Appropriate mechanisms address hardware resource scarcity, which renders the cloud expensive and inefficient. The Figure 1.2 illustrates the cloud computing primer. It consists of different implementation models, such as Private cloud, which is operated by a single entity or a private organization and it is not shared to the public or any third parties; Public cloud, which consists of publicly shared on-demand services and it is available for public; and Hybrid cloud, which is a mixture of private and public cloud, where an private organization or agency maintains the private cloud data and some of the data is discharged to the public cloud.

There is a security risk in the public cloud since it is a cloud of third parties that can be accessed by any user. The Figure 1.3 defines the form of services, the cloud computing offers. IaaS-Infrastructure as a Service deals with the services such as storage

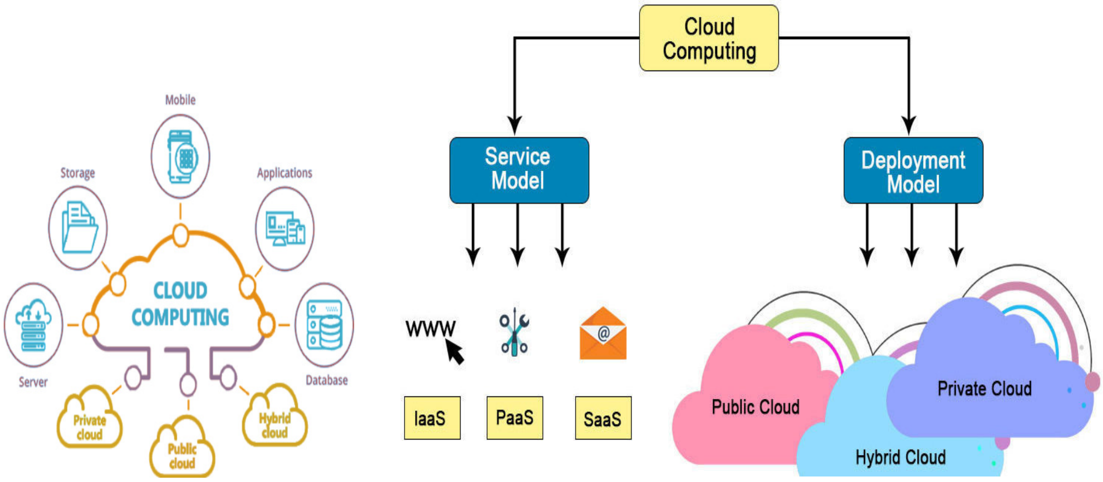


Figure 1.2 Cloud computing system and deployment models.

type, servers, and networks. Microsoft Azure, Amazon Web Service, and Google Computing Engines are the examples for IaaS. Similarly, PaaS-Platform as a Service offers different operating system platforms that deals with the Dropbox, Database Runtime Environments, and SaaS-Software as a Service deals with the Google applications.

Cloud Technology is basically a connection to different systems and their functionality through the LAN. Cloud customers need this connection from a collection of web services that maintain a collection of computing resources (Operating system, network, device, storage).

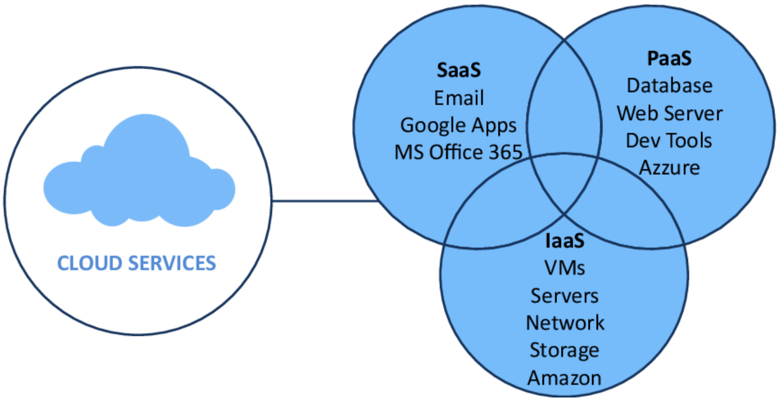


Figure 1.3 Types of cloud services.

Elasticity, Greater Storage Space, Elasticity, Trustworthiness, Geographically Dispersed, Dynamic Scalability, Flexibility, Less Management, Expert Service are the various functions of cloud Computing. Full resource usage, no direct spending on infrastructure, consumption-based charges, and execution time reduction are key business benefits of designing applications through cloud computing.Depending on the request from cloud customers and on production volumes, the implementation model are designed. Various cloud computing capabilities are summarized below:

- **Economical**: Due to the utilization pricing plan, cloud computing is very economical. There is no necessity for building a infrastructure for different services.

- **Increased storage space**: Storage and maintenance of large space is possible due to the significant infrastructure. Abrupt Server workload fluctuations are also handled effectively during the under and overloading situation.

- **Elasticity**: Cloud computing emphasizes that the most relevant building blocks necessary for deployment are used to bring cloud workloads or applications to the market quickly.

- **Legitimacy**: Cloud environment confirms the continuous operation without the interruption of machines, i.e., no data disruption, no code change during execution.

- **Geographically distributed**: Cloud customers may use services through a web browser, regardless of place, from where the users are accessing their website. Cloud data centres are geographically distributed.

- **Dynamic functionality**: Data and application resources for the customers are provided quickly with less response time.

- **Effectiveness**: The appropriate cloud provider ensures that the available services are continuously accessible.

- **Less management**: The cloud provider manages the infrastructure, software, and bandwidth. Hence, the customers need not manage any of the available resources in cloud environment.

- **Skilled service**: Cloud environment has skilled technical experts who manage different services. Cloud Storage services are regularly managed and operated at ease by trained data center professionals.

1.1 FEDERATED CLOUD ENVIRONMENT

Federated cloud refers to the broader number of geographically dispersed service providers that pool their servers to support the client request. Amazon, Microsoft, and Google are well known and large cloud service providers. The definition of the Federated cloud falls under the Infrastructure as a Service model, where the infrastructure is demanded by the customers which is available in the form of virtual machines.

The following are the significant entities that are active in a Federated cloud environment.

- **User**: A customer who demands a specific service is referred to as a cloud user/consumer. For example, the cloud services can be storage, networks or servers. The user or the customer submits the assignment without the knowledge of backend mechanism by requesting the services by paying for that specific service.

- **Public Cloud**: It is a free or pay-per-use services that is open to the public.

- **Private Cloud**: A single company manages the cloud, and the services are not open to the public. It is for a private organization only.

- **Hybrid Cloud**: It is a consolidation of private and public clouds. One or more corporate entities are involved in this type of deployment model.

- **Cloud Service Providers**: It provides services depending on the users request.

Amazon Elastic compute cloud, Google, Microsoft Azure are some of the leading source providers.

- **Workloads**: Cloud workloads are the loads produced by cloud infrastructures using a variety of applications and services. It is a combination of roles and activities presented in the cloud by the user/consumer.

For serving various purposes in different platforms, multiple clouds are used. The user application is supported by the public cloud deployment model if private cloud resources are unable to satisfy the request. To satisfy the customers request, service brokers may distribute different resources from geographically located service providers. Federated architectures are classified as follows:

- **Cloud Bursting Architecture**: When the customer runs out of internal cloud resources, user demand the public cloud service by paying for the services. Figure 1.4 shows the cloud bursting architecture. The user bursts the data into the public cloud during this process. The public cloud can be used as internal clouds without any network adjustment.

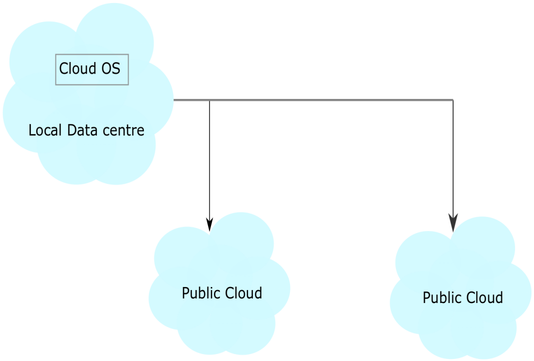


Figure 1.4 Bursting architecture.

- **Cloud Broker architecture**: The Cloud Broker is an interface between the provider and the user. Because of the terms and conditions, it is very hard for the customer to contact the provider. Therefore, a broker looks for the best providers to serve the request by hiding the management problems. The Figure 1.5 shows the cloud broker architecture.

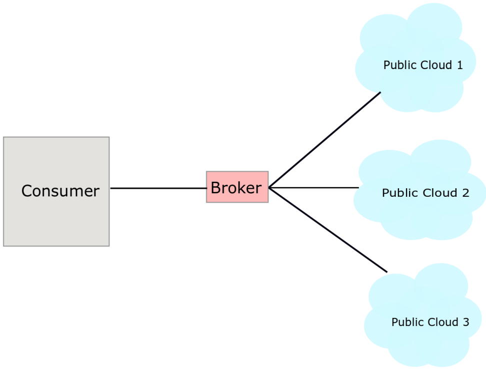


Figure 1.5 Broker architecture.

- **Aggregated cloud architectures**: The cloud is believed to provide the users with on-demand services on an infinite basis. The cloud cannot fulfill the requirements during the situations, where hardware or specialized resources are unavailable. Cloud providers then pool their services based on the structure and agreements and allow customer requests individually as shown in Figure 1.6.

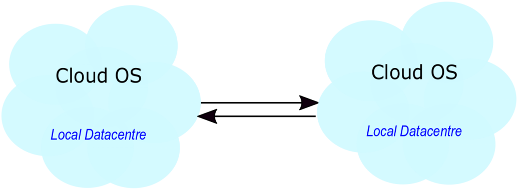


Figure 1.6 Aggregated architecture.

1.2 WORKLOAD CHARACTERIZATION

The workload is a collection of inputs provided to the processing infrastructure. Based on the successful processing of workloads, efficiency can be assessed. A different set of workloads have different characteristic. The term workload refers to all the inputs that a given technical infrastructure has obtained.Understanding the workload properties and behaviour is crucial for performance engineering studies dealing with the design and capacity planning of infrastructures and optimising their costs. More broadly, the user-perceived assessment of the QoS and QoE involves a thorough understanding of the underlying workloads. Similarly, the characterization of workload is the basis for creating efficient provision of resources, power management, strategies for energy efficiency, content delivery policies, protection frameworks, recommendation systems, and marketing strategies (Calzarossa et al., 2016). The models obtained from the method of characterization summarizes the key properties of the workloads and describe them. The amount of work that an organization handles provides an estimate of that entities productivity and output.

Components such as servers or database systems are openly allocated to an incoming workload. It is difficult to assess any model without understanding the workload. Analysis of the performance is carried out over time, relative to the workload that was anticipated. The workload can be strictly technological (e.g., supporting the operating system or running anti-virus software) or business-oriented (e.g., encouraging consumers to build documents by purchasing on an e-commerce platform or end-users). A workload can be broken down into various jobs, further splitting them into the number of tasks shown in the Figure 1.7. For example, workloads are raw inputs from the user to the server concerning the site, where the task is basic work to be performed, such as accessing the menu, password, etc. Workload based on different criterias are shown in the Figure 1.8.

Workloads are categorized based on the processing models, computing environ-

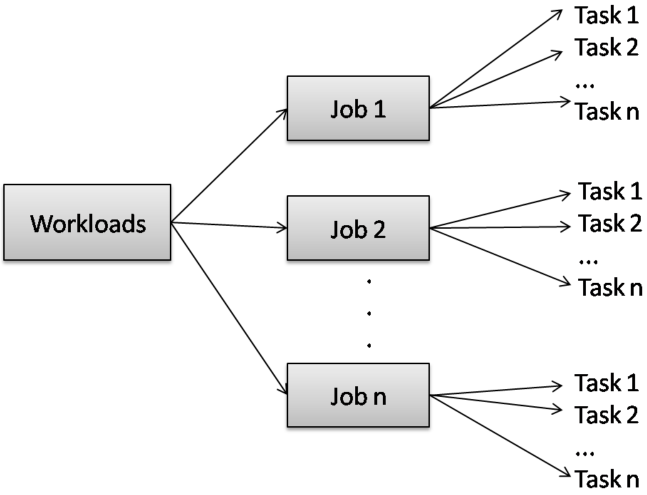


Figure 1.7 Distribution of Workloads.

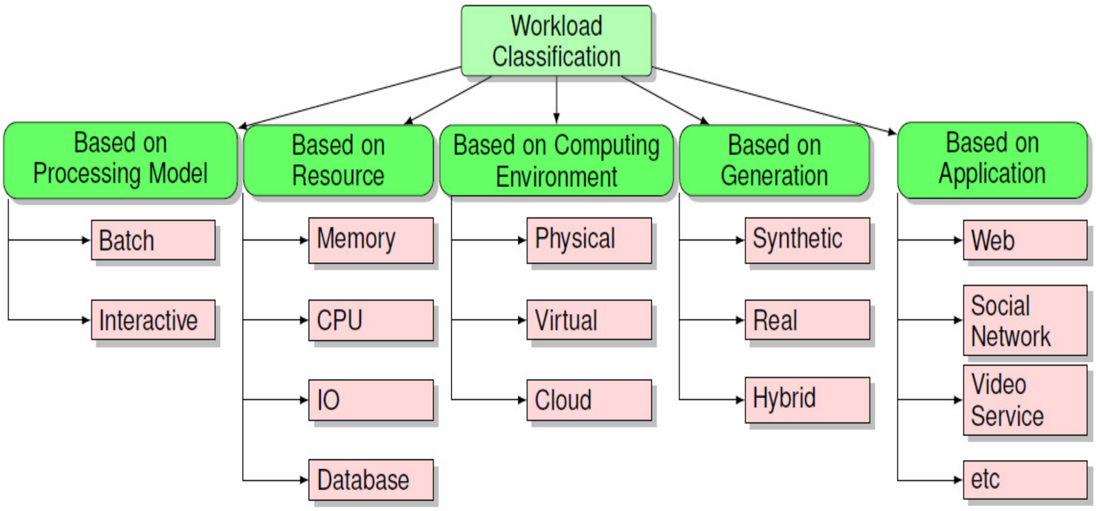


Figure 1.8 Classification of workloads on different aspects.

ment, resources and applications.

1. **Based on the processing model**: The workload can be classified into two categories: Batch workloads and Interactive workloads, based on the processing model.

   - *Batch workloads*: These workloads are intended for background service. Typical workloads by batch tend to handle enormous data volumes. These workloads require a lot of computation and storage. Many loads may consist of several repetitive transactions in a database, each with a heavy workload. A simple instance would be the measurement of the days gross revenue. In this instance, each day, sales transactions will be retrieved from the database by the batch process, recover the number of sales, and maintain a running sum.

- *Interactive Workloads*: An interactive workload is a request-serving application that runs indefinitely. The software accepts the user requests on an ongoing basis and handles the requests and responds to the user.

2. **Based on Resource requirement**: The workloads are classified into Memory, CPU, I/O and Database based on the resource usage.

   - *Memory Workload*: Each program or instruction requires some memory to store temporary or permanent data and conduct intermediate computations. The memory workload describes the memory consumption of the entire device over a given period or at a specific moment in time. A lot of virtual memory is used for paging and segmentation activities, thereby increasing primary memory usage. However, when the number of executed instructions is very high, the memory becomes a bottleneck for results. This means that more memory is needed or plans need to be more efficiently managed.

   - *CPU Workload*: The workload of the CPU shows the number of instructions executed by the processor over a certain period or at a specific time (Yang et al., 2003). Suppose the Processor is overloaded all the time or the processing power decreases, or the CPU consumption falls below a certain threshold. This signifies that the processing power increases. Further efficiency improvements can be achieved for the same number of instructions running on a CPU at a given time by reducing the number of cycles followed by adequate execution instructions.

   - Input/Output Workload: Most applications prefer to spend considerable time gathering input and processing output. Consequently, the workload of input-output (I/O) combinations on a system must be thoroughly analysed to ensure the required load efficiency specifications. A statistical number of inputs received by a system and the number of outputs produced by a system over a given period is called the Input-Output workload.

- *Database Workload*: It is possible to evaluate databases for their memory utilization, full load throughput, and I/O throughput. Each of these components can give a small approximation of the database output and its parameters. However, the real workload of databases can be analyzed by estimating the number of queries performed by the database at a given time.

3. **Based on the Computing environment**: Workloads are be executed in three fundamental environments; Physical, Virtual and Cloud environment.

    - *Physical*: It is in the form of dedicated servers, where the workloads are provisioned directly on the hardware.
    - *Virtual*: Workloads are delivered on a hypervisor that hides from the workload of the underlying hardware resources. This enables multiple workloads to easily share the physical resources and use it effectively.
    - *Cloud*: Workloads are supported either by third parties or run internally on massively vitalized systems, which could have thousands of underlying servers, storage, and network assets pooled together.

4. **Based on Generators**: Workload can be classified into Real and Synthetic loads based on the generation.

    - *Synthetic workloads*: The important requirement for the synthetic workload generator is that the workloads generated should reflect the real workloads and retain the important features of real workloads, such as inter-session and intra-session intervals. There are two approaches to the generation of synthetic workloads: 1) an empirical process in which traces of applications are sampled and replayed to create synthetic workloads; 2) an analytical system that uses mathematical models to determine the characteristics of the workload used by a synthetic workload generator.

   - *Real workloads*: It is beneficial to understand the characteristics of real workloads on a large production, but using synthetic workloads, it is difficult to extract the minute behaviors. Real workloads, such as Microsoft Azure traces, Google workload datasets and Yahoo traces are freely accessible for research purpose.

5. **Based on Applications**: Workloads are categorized into various applications, such as the web, social networks, video services, etc.

   - *Web workloads*: Workloads have greatly evolved in the functionality and design of the web. Conventional web workloads primarily consists of web or proxy server HTTP requests from clients to the websites. Page resources, traffic properties, access patterns, and user behavior relate to the typical web workloads.

   - *Workloads of online social networks*: The workload of online social networks refers to the load provided by the users to take advantage of the technology advanced services such as collaboration, networking, etc.

   - *Video Service Workloads*: The video service workload refers to the load created by the users who access and distribute the content either provided or self-generated by media producers. These workloads concentrate on the different features and forms of media and also the users experiences with the media.

   - *Mobile Device Workloads*: Mobile devices incorporate various mobile applications that allow users to access and share content and resources, creating some complicated interactions with their devices.

The workload classification is focused on the experimental methods based on observing the technical infrastructures as they run. Characterization of workloads have been widely researched for computers with many practical applications. Virtualized data centers workloads are primarily defined in terms of their utilization of resources. By concentrating on application characterization, user and virtual machine behavior, researchers have focused on determining the workloads processed by private and public cloud infrastructures.

1.3 ARTIFICIAL NEURAL NETWORK

A neural network is a sort of biological brain-inspired machine learning algorithm. It is a computer device designed to replicate how information is analyzed and interpreted by the human brain as shown in the Figure 1.9.

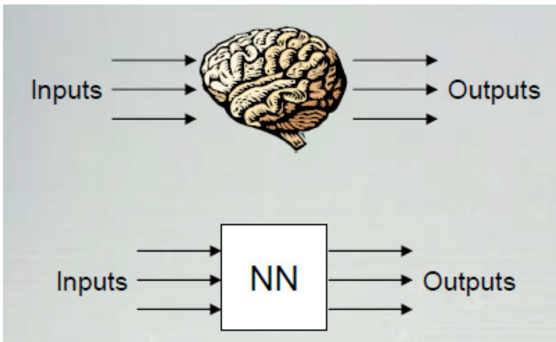


Figure 1.9 Neural network interpreted by human brain.

Neural networks have three main components, each component consists of nodes, input layer, a hidden layer, and an output layer, which is shown in the Figure 1.10.

ANNs are composed of numerous nodes that mimic the human brains biological neurons. The components of neural network are shown in the Figure 1.11. The neurons are connected by the links that communicate with each other. The nodes acquire the input data and carry out the essential data operations. The outcome is transmitted to other neurons. The output is called as activation or node value at each node. Each

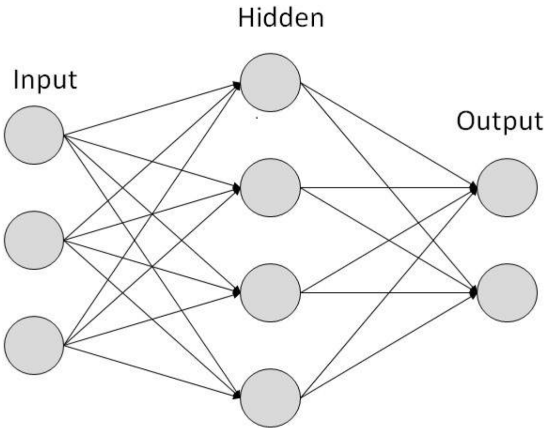


Figure 1.10 Basic Neural network layers.

connection is linked to a weight factor. ANNs are capable of learning by adjusting the weight values.

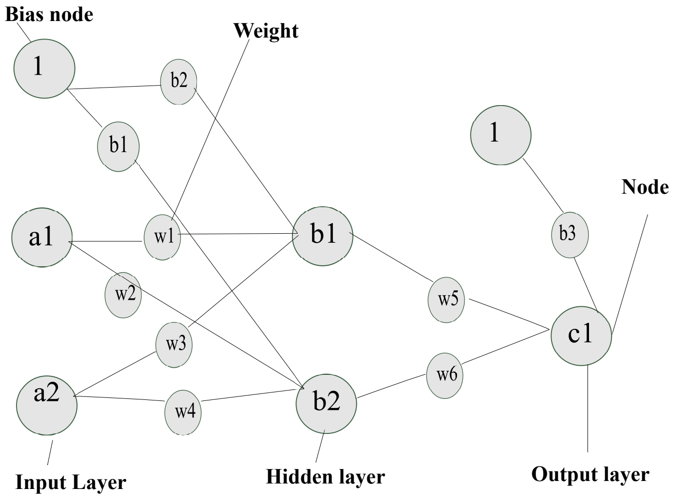


Figure 1.11 Components of Neural network.

There are two topologies of the Artificial Neural Network - FeedForward and Feedback

- **FeedForward ANN**: In this type of neural network, the flow of knowledge is one-way. A device sends information to another unit from which no notifications are obtained which is shown in the Figure 1.12. Feedforward loops are not accessible. They are used in generation/recognition/classification patterns. The inputs and outputs are defined at the initial stage.

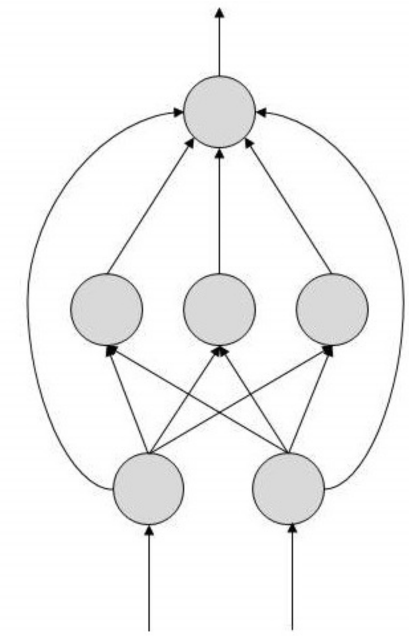


Figure 1.12 Feedforward Neural Network.

- **FeedBack ANN**: A device sends the information to another node and the notification will be recieved back. Feedback loops are permitted here as shown in the Figure 1.13 It is used in content-addressed memories.

Neural networks use three dataset types:

- **Training datasets**: It is used to alter the weight of a Network.
- **Validation dataset**: It is used to mitigate an issue called overfitting
- **Testing datadets**: it is used as a final test to determine how well the network has been trained.

Each arrow in the topology diagrams shows a relation between two neurons and offers a data flow pathway. Each link has a weight, an integer that regulates the signal between the two neurons. If the network produces a desirable performance, no weight

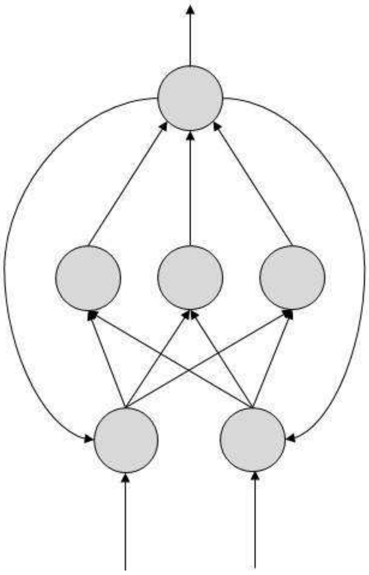


Figure 1.13 Feedback Neural Network.

adjustment is required. However, the machine switches weights to boost subsequent outcomes if the network produces a poor output or an error.

One of the essential options for creating a neural network is determining the activation feature to apply. Activation functions shown in the Figure 1.14 are the basis for the neural network to learn about complex and continuous interactions between variables. It makes the network non-linear. Every neuron (except the neurons input layer) calculates the weighted sum of inputs, adds a particular bias, and then uses an activation function.

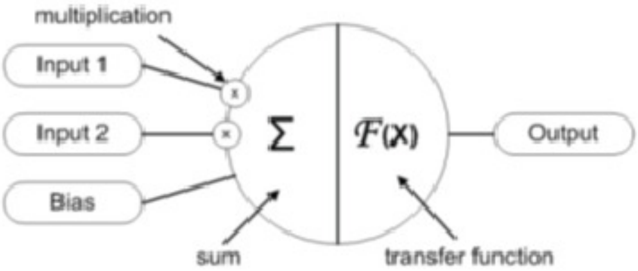


Figure 1.14 Activation function in a Neural network model.

A lot of activation functions are available. Commonly used functions are

1. **Logistic**: It ranges between 0 and 1. It is often referred to as the activation function of the Sigmoid. The weighted number of inputs are used here. It is defined in the Eq. 1.1
$$f(x) = \frac{1}{1+e^{-x}} \tag{1.1}$$

2. **Tanh (hyperbolic tangent)**: It is similar to a logistic activation. The performance varies from -1 to 1 on both sides of the zero axis with equal mass and is thus zero-centered. Tanh thus overcomes the logistic activation function's non-zero-centric problem. The optimization is, therefore, relatively simpler than the logistic function. It is defined in the Eq. 1.2
$$f(x) = \frac{e^{2x}-1}{e^{2x}+1} \tag{1.2}$$

3. **ReLu (Rectified linear units)**: The input data is the final result if the outcome value is positive, otherwise, the result output is zero. It is defined in the Eq. 1.3
$$f(x) = \begin{cases} 0 \quad for \quad x < 0 \\ x \quad for \quad x \geq 0 \end{cases} \tag{1.3}$$

1.4 META HEURISTICS OPTIMIZATION

Resource utilization is one of the important research problems that is focused on efficient results. It is better to find a suboptimal solution in the cloud world but in a short time. Metaheuristic methods have shown themselves to achieve almost optimal solutions for such issues within a reasonable time. There are two major categories of cloud computing providers: the first is the supplier of cloud services, and the second is the cloud user. Cloud service providers rent their services to cloud consumers, and cloud consumers submit their tasks to the resources. Providers are concerned with their applications efficiency, while the consumers are more interested in using their resources efficiently. This optimization calculation can, therefore, be divided into two types: Consumer-desired and Provider-desired. Below are such optimization parameters needed by the user and the provider when planning activities in a cloud environment:

1. **Consumer-Desired**:

    - *Makespan*: It indicates the finishing time of the last task. The most popular optimization criterions while scheduling the tasks is minimization of makespan as most of the users desire fastest execution of their application.
    - *Flow time*: It is the time at which all the tasks complete its job. Task must be processed in the ascending order to minimize the flowtime. Flow time value minimizing is the reduction in average response time of the schedule.
    - *Tardiness*:This defines the time elapsed between the deadline and finishing time of a task i.e. it represents the delay in task execution. Tardiness must be zero for an optimal schedule.
    - *Waiting time*: It is the time between the execution time and the submission time.
    - Turnaround time: This keeps track of how long it takes for a task to complete the execution since its submission.
    - *Fairness*: A desirable characteristic of scheduling process is fairness which requires that every task must share the same amount of CPU time and no task should be starved.

2. **Provider-Desired**:

    - *Resource utilization*: Another important criterion is to maximize the resource utilization i.e. keeping resources as busy as possible.
    - *Throughput*: It is defined as the total number of jobs completing execution per unit time.

- *Optimization*: It always has an objective and a constraint associated with it. The objective defines the best possible option whereas the constraint defines the restriction imposed. So, both optimization objective and constraints are related to each other. Following are a few common constraints considered while scheduling.
- *Priority constraint*: It represents the urgency of a task to complete at the earliest. Priority can be decided on the basis of deadline of a task, arrival time of a task or advance reservation. The tasks with shorter deadlines can be given higher priority and scheduled first.
- *Dependency constraint*: It represents the sequence of tasks based on their dependency. If there are precedence order among tasks, then it cannot be scheduled until the parent tasks are finished.
- *Deadline constraint*: This represents the time till which the task or the batch of tasks to be finished.
- *Budget constraint*: This represents the restriction on the total cost of task execution.

Following are the two important optimization techniques used in this thesis.

1.4.1 Genetic Algorithm

Holland implemented the Genetic algorithm in 1975 that is based on natural evolution. In GA, every chromosome is made up of a string of genes. The first population is randomly taken. A fitness is determined to verify the environmental suitability of the chromosome. The process is repeated until enough offspring is produced. The steps involved in GA cloud optimization algorithm is as follows:

Step 1: Initial population- a number of individuals to the problem.

Step 2: Fitness function- defines how an individual can fit (an individuals capacity to compete with other individuals).

Step 3: Selection- Choose two parent chromosomes according to fitness from a group.

Step 4: Crossover- Crossing the parents to form a new offspring.

Step 5: Mutation- Random modification of the chromosome, which also supports the idea of population diversity.

Step 6: Fitness- The fitness value of these newly formed chromosomes, known as descendants is determined.

Step 7: Repeat steps 3 to 6 until the stopping condition is met.

Step 8: Output- The best chromosome is selected as the final solution.

1.4.2 Artificial Bee Colony

It is inspired by the foraging behavior of honeybees, and a set of honeybees is called a swarm, which can perform tasks through social cooperation. The ABC algorithm has three types of bees: employee bees, onlooker bees, and scout bees.

Step 1: Initialization- set of the population for a given problem.

Step 2: Employee bees-Bees search for food around the food source; they share information with viewers about those food sources.

Step 3: Onlooker bees-From those gathered by the worker bees, bees tend to choose good food sources.

Step 4: Scout bees- These bees are transformed from a few employed bees who abandon their source of food and look for new.

Step 5: Fitness- The fitness value of the produced food supply is determined.

Step 6: Repeat steps 2 to 4 until the stop condition is satisfied.

Step 7: Output: The best solution (greater fitness value) is choosen as the final solution.

1.5 PROBLEM STATEMENT

To design and develop an effective workload management using prediction and optimization technique with the help of meta-heuristic approaches and computational intelligence for federated cloud environment.

Research objectives:

1. To develop a conceptual framework for workload management so that the considered workload is allocated efficiently to the respective resources.

2. To design a prediction model using feature extraction and Artificial neural network to improve the accuracy and efficiently manage the system specific workloads.

3. To design an optimization algorithm using Meta-heuristic based techniques to optimize QoS parameters in a federated cloud environment.

1.6 OUTLINE OF THE THESIS

The rest of the thesis is organized as follows.

Chapter 2 discusses the related work by reviewing the relevant works carried out by the researchers in the fields of workload characterization, prediction and optimization in the cloud domain in the context of the proposed approaches. The pros and cons of the related works are analyzed to identify the research gap.

Chapter 3 discusses about the conceptual framework designed that consists of set of modules which are inter-dependent on each other for workload management in a federative cloud environment.

Chapter 4 describes the novel fitness feature extraction model which extracts the features from the Microsoft Azure traces. The featured dataset such as CPU, Memory, Disk are considered for the prediction.

Chapter 5 describes the proposed prediction algorithm using Artificial neural network with the modified activation function. The prediction framework and the algorithm consisting of set up, neural network and output units are discussed here.

Chapter 6 discusses the proposed novel optimization technique consisting of Bee optimization and Genetic mutation algorithm. In addition, the proposed model testing and evaluation using the workload data traces from federated cloud service provider are discussed here.

Chapter 7 gives the conclusion of the thesis and future work.

Chapter 2

LITERATURE REVIEW

Due to the extreme dynamicity among the cloud workloads, characterizing becomes a tedious process. Most of the workloads arrive very frequently and exist over a short period. For retaining better performance, the workload prediction and optimization over three significant components is essential viz, utilization of CPU, availability of storage and space in memory and disk.

In this chapter, relevant research works in the areas of resource management in federated environment, cloud workloads, prediction models and nature inspired optimization techniques are analyzed, to understand the pros and cons of the approaches proposed by the various researchers. The research gap is identified which plays a pivotal role in the research work presented in this thesis.

2.1 OPTIMIZED MANAGEMENT OF RESOURCES IN FEDERATED CLOUD

In a federated cloud environment, there is one or more number of cloud service providers who share their servers to service the user demands. The service providers in a federated environment have their servers which are geographically distributed. The leading service providers are Microsoft, Amazon, and Google. Customers of federated cloud environment requests for the infrastructure as a service and it is served in the form of virtual machines. Cloud service providers have a huge number of resources for serving the client. The broker helps in monitoring, managing, accessing the cloud service provider irrespective of the geographically situated servers. Once the broker choose the best cloud service provider to serve a client request, the Service level agreement is signed between them to access the service. Hence, in this type of environment quality of service is high comparing to a single deployment cloud.

Over a single deployment model, the federated environment has lots of benefits which are as follows.

- **Scalability**: When the demand is high, cloud service provider shares their resources and provide services via service broker.

- **Multi-cloud Deployment**: It helps in reducing the cost required to serve the client request by aggregating the shared resources from the different providers.

- **Fault-Tolerance**: There will be data duplication, and hence no data loss exists during the natural disaster or downtime of servers.

- **Performance**: Due to the multi-cloud deployment, the cost is minimized, a Response time of serving the request will be low, which indirectly increases the performance measure.

Federated cloud consists of different levels of coupling including resource cooperation, monitoring, remote controlling, etc. (Moreno-Vozmediano et al., 2012) explained various levels of coupling in a centre. As given in the Table 2.1, coupling levels are classified as Loosely coupled, Partially coupled, Tightly coupled federative cloud instances.

- **Loosely coupling**: In this type of coupling, very less of inter-operation is done. Basic operations such as monitoring and controlling is done in this level. Advanced operations like migration are not included.

- **Partially coupling**: More than one number of cloud providers partner and share their resources with respect to their terms and conditions.

Table 2.1 Comparison on the types of coupling level

Level	Operations	Security
Loose	Basic functions on virtual machines	Single cloud in the organizations
Partial	Controlling and maintaining VMs	Agreements of framework
Tight	Scheduling to a particular resource, live migration	Consumer region sharing

- **Tightly coupling**: These types of coupling is done on inter-organization with the same architecture and OS type. Advanced features like Remote monitoring, VM migration is done here.

(Shishira and Kandasamy, 2020b) classified the federated architectures into different categories which is explained in the Chapter 1. Table 2.2 gives the comparison on the architecture of federated cloud.

Table 2.2 Comparison on the architecture of federated cloud

Architecture	Levels	Cloud Type	Benefits
Bursting	Loose	Bursting of workloads from private to public cloud	Helpful during resource ex-haustion
Brokering	Loose	Chooses among the best public service provider	Optimization of cost, execution time
Aggregation	Partial	Hybridization of public and private clouds	Resource sharing to meet the user needs
Multi-tier	Tight	Large cloud data centres combining public, private with several data centres	Scalability

Some of the challenging tasks in the federated environment are listed below. Aggregation of various cloud providers makes it a challenging task.

- **Portability**: It is very important to move the data safely on to different cloud data centers. Whenever there is demand from the client, it is very necessary to serve that particular request without any delay. Also, important to combine two or more private and public clouds to satisfy the business needs. Multiple clouds have to share their resources without violating the SLAs. The audit process is different with respect to the cloud providers.

- **Deployment plan**: Cloud providers offer a different set of resources based on the user demand. The service broker has to choose the best provider resources based on the application requested to meet its needs. But when there is an uncertainty in user requests, allotting the resources maintained by the cloud provider is a tedious task. It shouldn't go waste while in the off-peak demand.

- **Quality of Service**: Quality of Service plays a major role in serving the clients request. It is mainly based on the SLAs that are agreed between the provider and the consumer. Sometimes, if the resources are exhausted due to the natural disasters, it is important not to violate the SLA by compromising with the quality of service.

- **Consumer specific coercion**: User or a consumer can specify specific requirements to deploy their model on to the public cloud. During the process, the user can demand one or more VMs which may violate the conditions of the resource present in the data centers of one of the providers.

- **Jurisdiction**: Service providers have to adhere to the jurisdiction wherein the data centers placed is in specific regions. Thus, the client can deploy his/her model complying with his/her own regional laws.

- **Pricing**: Different service provider has different sets of pricing labels depend on the type of services provided. Amazon EC2 generally has three types of processing reserved, on-demand and spot type. Elastic hosts allow to customize the cloud instances based on CPU, disk, memory size. Providers charge based on network bandwidth, storage, memory.

2.1.1 Optimized Management of Resources

Management of resources refer to selection, monitor, increase or decrease the resources based on the demand. During off-peak hours, steady resource provisioning has no demand. Hence, automatic scaling of resources helps in auto increasing or decreasing the resources such as Infrastructure to the consumers.

In the cloud, both the provider and consumer try to obtain an optimal solution for resource provisioning. Provider tries to utilize the resources efficiently by not violating the QoS. While the consumer tries to minimize the service cost by deploying the applications and get the best service efficiently. Table 2.3 shows the existing frameworks proposed by the various researchers.

Table 2.3 Optimized scheduling of resources in federated cloud

Authors	**Framework**	**Parameter**	**Proposed model**
(Van den Bossche et al., 2010)	Bursting	Cost	Binary integer
(Javadi et al., 2012)	Bursting	Cost	Integer programming
(Chaisiri et al., 2011)	Brokering	Cost, response time	Stochastic method
(Tordsson et al., 2012)	Brokering	Cost, response time	Binary integer
(Lucas-Simarro et al., 2013)	Brokering	Cost, execution time	Binary integer
(Breitgand et al., 2011)	Aggregation	Energy consumption	Greedy method
(Vecchiola et al., 2012)	Aggregation	Performance	Integer programming method
(Wright et al., 2012)	Aggregation	Cost, execution time	Binary integer
(Calheiros et al., 2012)	Aggregation	Cost	Deadline management method

(Javadi et al., 2012) proposed a resource provisioning model for federated cloud. The author has used brokering method to make use of public resources while satisfying the user need by internally using the private cloud. (Van den Bossche et al., 2010) proposed a method to optimize the resource provisioning by bursting the cloud from the private to public clouds. (Chaisiri et al., 2011) used stochastic programming to optimize the cost using broking technique. (Tordsson et al., 2012) considered two types of resource provisioning, 1) to optimally place the VMs and 2) monitor and control the resources across different providers. (Lucas-Simarro et al., 2013) optimized the cost and performance using binary integer programming. Authors used different scheduling strategies to automatically scale the resources during the peak time. (Breitgand et al., 2011) used Integer programming system by efficiently serving QoS for load balancing. (Vecchiola et al., 2012) designed a model for the provisioning of resources from shared providers effficiently. (Wright et al., 2012) proposed a method for efficiently searching the best infrastructure for the cloud service providers to serve the client request. (Calheiros et al., 2012) presented an architecture from Intercloud to discover the best providers for the client request.

The outcome of the survey conducted which is presented in this section of the thesis are as follows.

1. **Consumer specific SLAs**: Cloud service provider provide SLAs depending on their resources and to benefit their performance and optimizing the resources. Based on the specific applications SLAs can be configured and benefitted as per the customer requirements.

2. **Consumer desired locality brokering**: Based on the locality and region which is profitable to consumer, cloud broker can optimize to choose the resources from specific providers.

3. **Resource scheduling in the federated environment**: Workloads can be predicted before going into the broker phase and the broker can choose the best

providers before handling a specific user request. Hence can optimize parameters such as delay, response and execution time.

2.2 WORKLOAD CATEGORIZATION IN COMPUTING

There is no commonly available definition of particular workload in the literature. The definitions given by several authors with respect to the computing environment is provided in this section. Also, classified the workloads based on the different processing models.The workload in computing refers to the capacity of the computer systems to operate. Table 2.4 provides the definitions given by various researchers.

Table 2.4 Workload Definitions provided in the literature

Authors	**Definition**	**Remarks/observations**
(Cetinski and Juric, 2015)	Workload of infrastructure resources can be presented as a time series, which is a sequence of data points typically measured at successive points in time spaced at uniform time intervals.	Workload is referred as Job
(Paton et al., 2009)	Workload is a set of workflows which is set of tasks	Granularity is set of tasks and queries
(Chen et al., 2012)	A workload consists of many jobs, each of which is represented as a directed acyclic graph (DAG). Each node in a DAG represents a task.	HPC Workloads were considered in the work
(Chang et al., 2014)	The amount of processing slot that the computer has been given to do in the cloud environment	Workloads is referred as Job
(Bahga et al., 2011)	Traces of events such as arrival of requests along with the time-stamps, details about the users requesting the services	Workloads are generated synthetically ad granularity considered is Task

A workload is a computing task that requires access to a mix of resources. Though there are different definitions provided in the literature, the workloads are majorly referred to either different sets of task or jobs. Thus, the workload can be broken down into different types or sets of jobs which further splits into number of tasks. For example, with respect to the web, workloads are complete inputs coming from the user to the server, where job is particular work to be done such as accessing the menu, login etc. Tasks are a lot simpler than jobs which specifies some of the navigations involved during the course.

2.2.1 Characteristics of Workloads in cloud

To distinguish between different workloads, we considered a conventional web server and a cloud server from the literature.

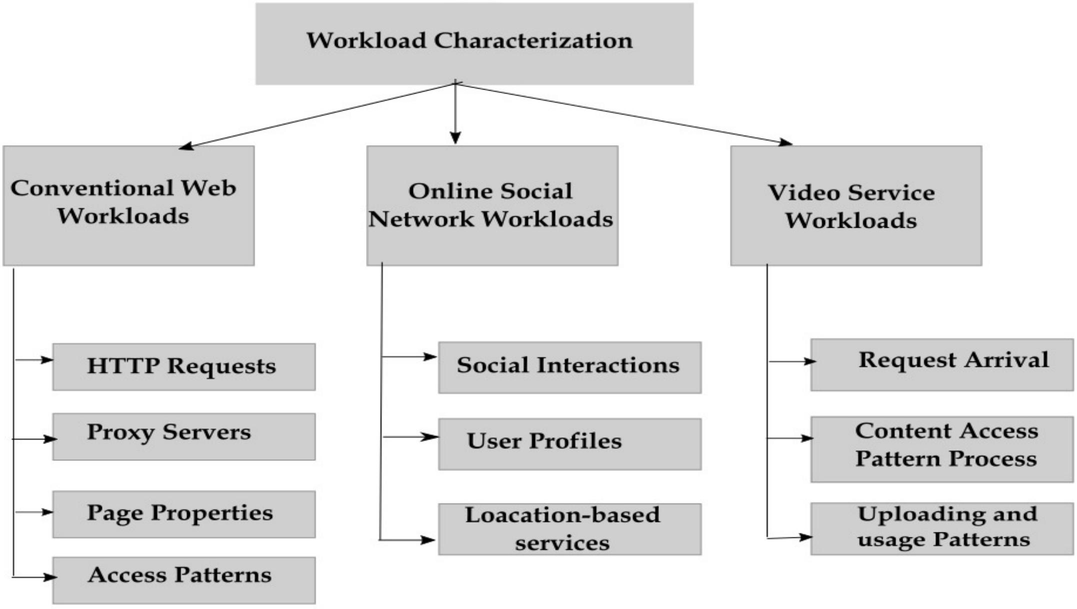


Figure 2.1 Workload characterization in Computing

In the Figure 2.1 characterization of the workloads associated with popular application domains are described in terms of data collection and data analysis. The term workload characteristics refers to the demands placed by various system resources. Each workload component is described by a set of parameters which explain these demands. The Different workloads have different characteristics and the best platform for a particular workload to run on depends on the nature of the specific workload. Workload characterization relies on experimental approaches based on the analysis on the technological infrastructures while they are operating.

Characterization of computer workloads has been extensively studied with many practical applications. (Zhang et al., 2009) created and used a set of fixed traces to create profiles of workloads based upon machine utilization and wait times. Table 2.5 provides different characteristics of application workloads.

Table 2.5 Characteristics of Workloads in general

Applications	**Characteristics**
Conventional Web Workloads	HTTP requests, Proxy servers to download pages, page properties, traffic properties, access patterns, user behavior etc.
Online Social Network Workloads	Social interaction, User profiles and activities, micro blogging services, location-based services
Source code level workloads, Micro architecture dependent and independent	Source code-statement distribution in programs, distribution of operations, Machine-dependent-cache hit ratios, bandwidth consumption
Data mining workloads	Memory Characteristics- Size of input record, size of prototype array, stack distance for each memory
Proxy server workloads	Number of requests by clients, Response time(Time, Size , throughput)
Server Workloads	Disk storage and communication/ traffic parameters
Decision support system(DSS) Workload from Transaction Processing Council(TPC)	Queries and database sizes
Java server benchmarks	high instruction cache, Branch Target Address Cache miss rates
e-commerce application server	CPU instructions and performance metrics
Video Service Workloads	Request arrival process, content access patterns, uploading and usage patterns

In the literature, the characterization of cloud workloads focuses on two main targets, namely jobs/tasks and Virtual Machines (VMs). In the Tables 2.4 and 2.6 Different authors have given a different description of workloads and its characterization.

Thus, based on the different viewpoints on workload characterization, one of the promiscuous characterizations we followed is given in calzarossa.

Table 2.6 Workload Characterization in cloud environment

Sl No.	Authors	Generators	Considered Characteristics
1	(Cetinski and Juric, 2015)	Real+ Modified	Submission time, wait time, run time and the number of processors used for each job
2	(Yuan et al., 2016)	Synthetic	NA
3	(Chen et al., 2012)	NA	NA
4	(Roy et al., 2011)	Real(Soccer worldcup)	NA
5	(Kang et al., 2011)	Synthetic	Scheduled times, CPU block wait, running time
6	(Mulia et al., 2013)	NoA	Arrival and running time
7	(Zhang et al., 2009)	Synthetic	Arrival rate of workloads
8	(Morariu et al., 2012)	Synthetic	Lab activities(Scheduled time, Finish time)
9	(Zhang et al., 2014)	Real(Yahoo)	Video stream requests time
10	(Bahga et al., 2011)	Synthetic	Inter-session interval, Think time, Session length
11	(Marcus and Papaemmanouil, 2016a)	Real (Google trace)	Arrival and Submission times

As stated in (Calzarossa et al., 2016) Cloud workload characterization can be analyzed by focusing on

- virtualized data centers,
- cloud infrastructures,
- desktop and storage services.

The workloads of virtualized data centers are mainly characterized in terms of their resource usage. Numerous papers focus on the characterization of the workloads processed by either private and public cloud infrastructures by focusing on aspects, such as: application characterization, user behavior, and VM behavior. The majority of these studies rely on public anonymized trace logs made available by the providers (e.g., Google), whereas only a few studies analyze private data.

In terms of resource utilisation and network traffic, the workloads of evolving Cloud solutions with desktop applications and personal storage resources are analyzed. In particular, desktop workloads are investigated in (Kochut and Beaty, 2007) by focusing on the resources consumed by applications, the activities performed by the users, and the processes executed by the operating systems. The study shows that resource usage patterns are rather bursty. A consolidated cloud workload is therefore analyzed by aggregating the resource usage of individual desktops.

(John et al., 1999) describes the source code level workloads and its characterization with its objectives. The characteristics of workloads were Source code-statement distribution in programs, distribution of operations, Machine-independent static and dynamic size, computation to communication ratio etc, Machine-dependent-cache hit ratios, bandwidth consumption.

(Kim et al., 1999) considered data mining workloads that contains memory characteristics such as size of input record, size of prototype array, stack distance for each memory. The objective of the work (Murta and Almeida, 1999) is to analyze the web proxy cache logs containing proxy server workloads in which the Number of requests by clients, Response time (Time, Size, throughput) were taken into consideration.

(Boyd and Recio, 1999) identified and summarized the I/O system characteristics consisting of Server Workloads(On line transaction, e-business) with Disk storage and communication/ traffic parameters. (Clark et al., 2001) Characterization of TPC-H Queries on AMD AthlonTM Microprocessors for Decision support system(DSS) Workload from Transaction Processing Council(TPC) containing Queries and database sizes.

(Seshadri and Mericas, 2001) used Java server benchmarks containing high instruction cache, and Branch Target Address Cache miss rates. (Cain et al., 2001) used Java application workloads for e-commerce application server consisting of CPU instructions and performance metrics.

(Nogueira et al., 2002) showed a methodology for workload characterization for analyzing and understanding P2P networks for workload generated by the P2P network.

To know the properties of workloads in depth, characterization of computer workloads has been extensively studied with many practical applications.

We were able to find some of the issues related to the workloads in the cloud environment which are as follows.

1. **Characterization attributes**: Several authors considered the workloads based on different attributes such as Submission time, finishing time, inter-session interval etc. Yet, there are different types of attributes which can be considered during the processing of workloads. Also, in the literature, authors considered application specific attributes, hence workload parameters can be adaptive so that it can distinguish with respect to attributes.

2. **Granularity**: Workload is mainly referred to as either jobs or task in the literature. It can be made more specific and combine both for the further process.

2.3 PREDICTION MECHANISMS IN CLOUD ENVIRONMENT

Several researchers have worked on cloud workload prediction techniques of which static and history-based are well known. There is no domain-specific definition for workloads. Workloads are considered as the combination of jobs which are subdivided into many tasks at the granular level(Shishira and Kandasamy, 2020a). Due to a greater number of attributes, Deep Belief Network(Qiu et al., 2016) did not achieve efficient predictions. Reinforcement technique(Amiri and Mohammad-Khanli, 2017) for workloads prediction was designed which had a scalability issue. In short, the prediction accuracy result is mainly depending on the similarity between the test and the training phase.

The following are the three different methods where the researchers have addressed the efficient resource utilization problem.

- **Statistical techniques**: For short term predictions of time series loads, statistical techniques were used. Pattern matching between past occurrences and short-term load history is identified. Some of the methods include Autoregression model(Lin et al., 2011), Monte Carlo(Vercauteren et al., 2007), Moving average(Ardagna et al., 2012), Quadratic regression(Sun et al., 2013), Hidden Markov model(Khan et al., 2012). Sarikya et al.(Sarikaya et al., 2010) proposed a solution for the prediction approach problem by applying Kalman filters for the grid platform. Thus, statistical methods techniques are well suitable for short term predictions of time series data. For efficient data prediction, complex techniques such as machine learning methods are used.

- **Learning techniques**: As mentioned in the previous method, statistical techniques have a disadvantage in predicting the series data which contains large number of fluctuations and accuracy with the high error. Thus, machine learning methods such as k Nearest Neighbors, Support vector machines, Neural network approaches came into existence. Some of the complex learning methods such as neural networks give the best accuracy, but it is time consuming due to the number of layers in it. kNN is used by several researchers for pattern identification.

- **Hybrid Techniques**: Many researchers have combined statistical and learning techniques to achieve efficient prediction. (Zhang et al., 2019) proposed a framework for modeling workloads. To predict the workloads on the cloud platform, (Islam et al., 2012) used Neural networks and regression models.

The prediction of workload is very significant to offer a better ratio on the cost to benefit (Sahi and Dhaka, 2016). Several researchers worked on predicting the workload in cloud with many techniques of which the history-based and homeostatic prediction were well known (Yang et al., 2014). The accurate Prediction Model (PM) was proposed with Deep Belief Network (DBN) that resulted in inaccurate prediction due to the usage of many parameters during the training phase (Qiu et al., 2016). A reinforcement learning-based PM was implemented to forecast CPU demand with the established policies on physical machine quantity. The approach is beneficial as there was no need for any domain knowledge, but the drawback was its inadequacy in scalability issues (Amiri et al., 2017).

The usage of Long short-term memory (LSTM) in the prediction of workload was pervasive in different forms. A PM with Univariate LSTM was proposed to predict the workloads in cloud through CPU performance. However, it did not include either the memory or disk space usage (Song et al., 2018). A Bidirectional Long short-term memory (BLSTM) model was proposed for predicting the workloads in cloud through the utilization of CPU, memory, and disk space along with several other factors in both forward and backward propagations. The issue with the BLSTM is, it consumed more learning time (Gupta and Dinesh, 2017). Hence, PM was developed by integrating both LSTM and BLSTM, which reduced the learning time with an accuracy near to BLSTM (Gupta et al., 2017).

A Linear Regression (LR) model was used to predict the flow of requests in each cloud. Based on the fluctuations of workload, the PM changes itself over the regression model parameters (Amiri and Mohammad-Khanli, 2017). In short, the accuracy of the prediction methods largely depends on the similarities in application behavior in testing and training phases, and influences the result reliability. An encoder-decoder network was proposed based on GRU model to improve the RNN (Recurrent Neural Network) capability for prediction. The process of encoding was performed over the historical data to generate the vectors of fixed length and these vectors are decoded for predicting the future workload (Karimi et al., 2017). Another RNN-LSTM model was proposed to predict the workload based on the user request logs. The proposed model has the capability to generate the request sequence of user (Huang et al., 2017).

Thus an improper workload prediction leads to a substantial deviation in the estimation of resource demand that may result in resource under or over-provisioning (Xiao et al., 2012). Many researchers carried out several works in predicting the workload to optimize the operations in the cloud environment. The traditional methods applied for workload prediction provides sufficient accuracy but largely depend on the collected data. The machine learning techniques provided in the literature enhanced the accuracy in prediction. However, the data are considered to be independent which failed to establish the relationship among the workloads (Zhu et al., 2019). Also, the models provided in the literature improved the performance but, are not adequate, especially in the federated cloud environment. Additionally, the performance of the prediction model suffers due to the Gradient Descent and Newton method problem while predicting the workload in the cloud environment.

2.4 META-HEURISTICS OPTIMIZATION TECHNIQUES

Meta-heuristic optimiation provides near optimal solutions within the reasonable amount of time. As an outcome of this survey, we have categorized the studies under: Cost optimization, workload-aware optimization, green-aware optimization, meta-heuristics optimization.

The survey results are as follows.

1. **Cost optimization**:

   (Urgaonkar et al., 2015) developed an approach to solve Markov Decision Problem. This method was applied for scheduling in edge clouds. The objective of their work was to optimize the operational costs while providing the guarantee in performance.

   (Van den Bossche et al., 2013) proposed a scheduling method for deadline constrained workloads. Binary integer model was proposed for evaluating the computational cost. It was found that the method worked good in public but was less efficient in Hybrid Cloud. Their method was able to minimize the computation cost while scheduling.

   (Marcus and Papaemmanouil, 2016b) introduced a framework for workload management solution known as WiseDB. This approach helps in training decision tree models for query placement, scheduling and resource allocation. The referred paper states that their system can adapt its offline model with minimal re-training. Result show that their approach offered low cost strategies for different performance metrics and workload characteristics. Work can be extended by considering multi-metric performance goals that combine workload and query level constraints as well as dynamic constrains.

(Qiu et al., 2013) proposed a model to characterize the scheduling of heterogeneous workloads. They have proposed an algorithm for Virtual Machine allocation and task outsourcing. By using Lyapunov framework, work shows that the algorithm achieved the reduction in task outsourcing cost. Work can be implemented in real-time systems for further evaluating its performance.

(Yuan et al., 2016) presented total cost optimization problem. The objective of the work was to optimize the number of Servers by using cost aware workload scheduling method. Hence, the experiment shows that the total cost was reduced and throughput was increased. This work can be extended by considering memory and storage metrics.

2. **Workload-aware optimization**: (Garg et al., 2011) presented resource allocation problem within data centre. The main objective was to maximize the resource utilization and profit ensuring user SLA requests. It shows that, their mechanism reduce the server utilization by 60% over consolidation and migration. For Future work, different SLA's and penalty can be considered with the mix type of workloads for better provisioning of resources and utilization.

3. **Green-aware optimization**: (Chen et al., 2012) proposed a optimization algorithm for workloads which is used for minimizing the brown energy consumption across geographically distributed data centres. Experiments were conducted with the real workload traces which demonstrate that consumption of brown energy was greatly reduced up to 40% when compared with the other scheduling policies. The work can be extended by considering the cost analysis and also other workloads.

4. **Metaheuristics optimization**: (Zhang et al., 2014) introduced a ordinal optimization method to obtain the optimal solution in limited time frames. The Evolutionary approach was compared with the Monte Carlo and BlindPick method. Results show that the method achieved higher throughput in real world applications.

   (de Oliveira et al., 2013) proposed a scheduling algorithm for bio-informatics workloads based on the load balancing Ant Colony Optimization called as ACO Scheduling. This algorithm was used to find the best cloud in the federated environments for the efficient distribution of tasks. Experiments show that the algorithm is able to archive minimum response time to execute the task. The other features of bioinformatics applications that relies on scheduling such as CPU or I/O bound can be considered.

   (Morariu et al., 2012) presented a model and a scheduling mechanism for e-learning workloads using meta heuristic genetic algorithm. Referred paper shows that their scheme was able to achieve near optimal solution and also co-scheduling of the workloads.

   (Nan et al., 2013) presented the workload scheduling schemes which are used for multimedia clouds where minimizing the response time and cost of the resources were major objectives. Greedy scheduling method was used to schedule the workloads in the multimedia cloud. Simulation demonstrated that their scheduling schemes optimally balance the workload to optimise the response time and resource cost in multimedia cloud environment.

2.5 QOS PARAMETERS CONSIDERED IN DIFFERENT WORKLOAD ALLOCATION APPROACHES

Optimal cost scheduling methods aims at minimizing the operational cost and improve the cloud providers profit.
Some of the notable approaches are (Marcus and Papaemmanouil, 2016b) to recom-

mend cost-efficient workload scheduling strategies; (Yuan et al., 2016) to reduce the number of active servers through cost-aware workload scheduling.

Meta-heuristics approaches which uses evolutionary approaches (Zhang et al., 2014), ant colony (de Oliveira et al., 2013), and genetic algorithm (Morariu et al., 2012) are well known. In general these methods aim at increasing the throughput and decreasing the makespan. Profit and penalty awareness is considered in Li et al. (Li et al., 2012), and energy and deadline-awareness is considered in (Gao et al., 2013).

Major optimization objective considered is Cost. Optimizing the cost in cloud environment is very important to gain performance efficiency. In general there are user centric costs which need to be minimal, and Provider centric costs where gaining a high profit is an advantage. Other than this metric, there are several other parameters considered such as SLA Violations, Fairness, Load balancing, Energy consumption, Resource utilization, Scalability, Makespan, Throughput, Response time, Execution time, and very less on Reliability and Trust. SLA Violation, which is a commitment between a service provider and the consumer is a necessary Quality of Service parameter which includes the type of services to be provided, deadlines etc. Fairness is a desirable characteristics in which every task should get the equal share of CPU time and none of the tasks be starved. Fairness should be high for the efficient performance in the cloud environment.

(Kang et al., 2011) considered the efficient fairness across multiple clusters reducing the cost. Load balancing is one of the QoS parameter where every resources get equal amount of tasks and it reduces the cost of maintaining the systems and maximizes the availability of resources, hence it should be high. The Energy consumption is a key issue in operation and maintaining the datacenters that need to be reduced without degrading the performance and violating the SLAs. Another important parameter is maximization of resource utilization i.e. keeping the resources as busy as possible. This criteria is gaining significance as providers want to earn more profit by renting very less number of resources. The ability of the system or the network to handle the increasing load is known as scalability and in cloud environment it is necessary to have a high scalable systems. Makespan indicates the finishing time of the last task. Most of the users desire fast execution of their application hence makespan needs to be minimized. Throughput is the total amount of jobs that finishes the execution in given amount of time. For an efficient performance, more number of jobs need to executed in specific time, Hence the given throughput must be high. The time between the deadline and the finishing time of a job is response time which should be minimum for an optimal schedule.

A multi-objective GA algorithm was presented to discover community complex network structure that optimizes two objective functions to maximize both the intra-connections and minimize inter-connections among diverse communities (Pizzuti, 2011). Modified GA was provided for multi-objective task scheduling through different computing system, and the investigational outcomes showed superiority in schedule with the suggested algorithm than others (Sathappan et al., 2011).

A novel algorithm of Artificial Bee Colony (ABC) was presented for a multi-objective optimization problem. The Pareto dominance concept was employed to control the bee flight direction and the non-dominated solution of the vector. The suggested approach was extremely competitive and a feasible alternative for solving problems in multi-objective optimizations (Zou et al., 2011). An ABC decomposition algorithm (Zhou et al., 2018) was proposed to solve Many-Objective Optimization Problems (MaOPs) by transforming it into many subproblems that were concurrently optimized. The ABC algorithm was beneficial when resolving problem in scalar optimization with fast convergence speed.

Some other novel classification algorithms are proposed by integrating neural or deep networks with optimization techniques for the effective classification of the data. Deep Belief Network (DBN) was integrated along with the Particle swarm Optimization algorithm for effective image segmentation over the gold immunochromatographic strip (GICS). The segmentation accuracy is about 99% over the different images (Zeng et al., 2019). Another integrated algorithm of PSO and cellular Neural Network is involved in segmenting GICS. The proposed model provided higher accuracy than the existing algorithm over the peak signal to noise ratio (Zeng et al., 2014). While in some cases the optimization algorithm can be modified to estimate the effective parameters in the process and support in increasing its performance (Zeng et al., 2016).

2.6 BIO INSPIRED OPTIMIZATION ALGORITHM

Ant colony optimization is used to find the shortest path between ant colonies and a source of food. This novel approach was introduced by (Dorigo and Blum, 2005). To improve the performance of ACO and to make it more efficient, pheromone updating strategies are proposed (Sun et al., 2013) and (Mathiyalagan et al., 2011). (Liu and Wang, 2008) presented a task scheduling algorithm for grids by adaptively changing the value of pheromone. (Bagherzadeh and MadadyarAdeh, 2009) have introduced the concept of biased initial ants to improve ACO. Authors have also considered standard deviation of jobs in addition to pheromone, heuristic information and expected time to execute a job on a given machine.

The potential local search strategies are based on finding a resource for the problem

(Kousalya and Balasubramanie, 2009). (Chiang et al., 2006) have incorporated local search strategy at the end of each iteration to improve each obtained solution.

(Chen et al., 2009) addressed the time-varying scheduling problem based on ACO approach intended to minimize the total cost in a period while meeting the deadline constraint. For this, integrated heuristic is designed based on the average value of cost heuristics and deadline heuristics. The fitness value is evaluated by considering its performance in different topologies in a period. ACO can be improved by using knowledge gained from predetermined number of best solutions of previous iterations (Xing et al., 2010). The concept of knowledge matrix is integrated with the ACO algorithm. Knowledge matrix is changed by two methods, one is by knowledge depositing rule and other is by knowledge evaporating rule.

(Wen et al., 2012) proposed that ACO algorithm can also be combined with other algorithms such as Particle Swarm Optimization to improve its performance. It not only enhances the convergence speed and improves resource utilization ratio, but also stays away from falling into local optimum solution. (Pacini et al., 2014) addressed the problem of balancing throughput and response time when multiple users are running their scientific experiments on online private cloud.

Particle Swarm Optimization (PSO) is a computational technique introduced by (Kennedy and Eberhart, 1995) in 1995. Small Position Value (SPV) rule is one of the immensely used techniques here. Integer-PSO is the technology that surpasses the SPV when there is a significant difference in task duration and resource processing speed. (Liu et al., 2010) used fuzzy matrices to represent position and velocities of particles in which element in each matrix signifies fuzzy relation between resource and job i.e. the degree of membership that the resource would execute the job in the feasible schedule solution space.

(Kashan, 2009) proposed a metaheuristic algorithm termed as League Championship Algorithm (LCA) in 2009 which is used for global optimization. It is inspired by the contests of sport teams in a sports association (league). (Sun et al., 2004) used this algorithm for solving optimization problems related to cloud scheduling. Main aim was to minimize the makespan of the task in Infrastructure as a Service (IaaS) cloud. Based on the survey, following observations were made.

- A metaheuristic algorithm can be improved by combining it with other population-based metaheuristic algorithm or some local search-based metaheuristic algorithm. Advantage of combining two popular based metaheuristics is that the short comes of one algorithm can be overcome by strengths of other algorithms.

- Quality of solution can be improved by modifying the Transition/Fitness function in the given optimization technique.

2.7 COMPARATIVE STUDY OF CLOUD SIMULATION TOOLS

It is difficult for the developers to do an extensive research on all the issues in real time as it requires infrastructure which is beyond the control, also network condition cannot be predicted. Hence simulations are used which imitates the real time environment. There are various simulators developed for the research as it is difficult to maintain the infrastructure on premise. Thus to understand the tools in deep, we focused on five open source tools such as Cloudsim, CloudAnalyst, iCancloud, Greencloud and CloudSched.

This section briefs on the related work on the different simulators developed for cloud computing. Gangsim tool is introduced by (Howell and McNab, 1998) for grid computing. Gridsim toolkit is developed for modelling and simulation for grid computing by (Buyya et al., 2009). Calheiros et al. Compared different scheduling algorithms at the application level on the proposed tool (Calheiros et al., 2011). (Sakellari and Loukas, 2013) provided a survey on various mathematical model approaches which is helpful for the researchers for further simulations and implementation of the particular modelling techniques. (Youseff et al., 2008) designed a simulation-based cloud resource management model which focuses on dynamic service composition. (Huu et al., 2013) proposed a scheme constituting a model and an experimentation for energy data centres. (Guérout et al., 2013) provided a survey on energy aware simulators and techniques with the help of Dynamic Voltage and Frequency Scaling. CloudSched tool has been developed by (Tian et al., 2013) which is cloud simulation tool for virtual machines in cloud data centres.

Based on the given configurations, CloudAnalyst gives the best scheduling results among the consumer groups. CloudAnalyst is an extension of cloudsim which has a improved GUI feature in it. Both Cloudsim and CloudAnalyst are implemted on Gridsim and SimJava which considers the cloud centre as a huge pool of resources with variety of workloads. GreenCloud has been developed by (Hongyuan et al., 2006) at a package level for energy-aware in the cloud data centres. (Tian et al., 2011) developed a tool called iCanCloud that is favoured for the cloud infrastructure which has been implemented in C++, and the proposed tool was compared with the Cloudsim for its performance.

Cloud tools are divided into different categories based to their features.

Open source simulation tools namely Cloudsim, iCancloud, Greencloud, CloudAnalyst, CloudSched have been considered in the previous work which is shown in Table 2.7.

- Platform: Cloudsim, CloudSched, and CloudAnalyst are implemented in java, thus they can be executed on any machine. Green cloud is implemented in NS2 simulator and iCancloud in OMNET.

- Programming Language: The Programming languages are meant to their respective platforms. Cloudsim and CloudSched are implemented in Java, whereas tools such as Greencloud is the combination of C++ and OTcl, and iCancloud is implemented in C++.

- Availability and Graphical support: All the simulators discussed are freely available for the users. Except CloudAnalyst all other tools do not support GUIs. However, there is no full support provided in the CloudAnalyst. Hence it is mentioned as limited during the comparison.

- Parallel experiments: It includes the combination of different machines working simultaneously to process the task. iCanCloud is one such tool which helps in parallel experiments, and other simulator do not support this feature.

- Energy consumption model: Energy consumption model is used to compare different scheduling strategies which is helpful and efficient. Except iCancloud and CloudAnalyst other simulators support energy consumption model.

- Migration algorithms: Migration algorithms are helpful when there is a load in on premise datacentre and needs to be offloaded to the public or private centres for saving the total energy consumption or to improve the utilization of resources. CloudSim, CloudSched and CloudAnalyst algorithms are helpful in migration while others do not.

2.7.1 Comparison on simulation procedure

We have divided the simulation process into three different categories.

- Generating requests: Generation of customer requests varies according to the simulator. Cloudsim, CloudSched, CloudAnalyst generate the requests as virtual machines instances and put into waiting queue. Greencloud produces workloads and iCancloud uses jobs which is then added to a waiting queue which is to be executed.

Table 2.7 Comparison Guideline for Simulators

Items	**Cloudsim**	**CloudAnalyst**	**iCancloud**	**Greencloud**	**CloudSched**
Platform	Any	Any	OMNET	NS2	Any
Program language	Java	Java	C++	C++/ OTCL	Java
Availability	Open source	Open source	Open source	Open source	Open source
Graphics	No	Yes(limited)	No	No	No
Parallel experiment	No	No	Yes	No	No
Energy consumption	Yes	No	No	Yes	Yes
Simulation time	Seconds	Seconds	Seconds	Tens of minutes	Seconds
Memory space	Small	Small	Medium	Large	Small

- Data centre initiation: Datacentre provides resources. All the five simulators discussed are similar in initializing the data centre and offer resources such as memory, storage etc.

- Defining allocation: It describes scheduling which includes where and how the request is allocated and processed. Cloudsim, CloudAnalyst, iCancloud implement First Come First Serve allocation policy. Cloudsched adopts load balancing policies, whereas Greencloud implements Dynamic voltage frequency scaling to allocate the request.

- Output the result: Results are gathered for evaluation of the performance. CloudAnalyst uses limited graphical user interface which enables user to setup the experiment quickly.

2.7.2 Comparison of performance metrics

There are various different metrics for load balancing, energy efficient goals. Five simulators which we have discussed here use different metrics. Table 2.8 summarizes different metrics, objectives and the simulators which adopt these metrics.

Table 2.8 Comparison Guideline based on metrics

Metrics	**Optimization objectives**	**Simulator tools**
Average resource utilization	Maximize Resource	All Five
Total number hosts needed	Maximize Resource	All five
Average CPU utilization	Load balancing	All five
Make span	Load balancing	CloudSched

1. Resource utilization:
   - Average Resource utilization: Resources such as CPU, memory, harddisk can be computed and used.
   - Total number of hosts used: It describes the utilization of a cloud datacentre.
2. Metrics of load-balancing
   - Utilization of a single CPU: The observed time at which the average load is on a single CPU.
   - Makespan: It is defined as the duration of the execution time on all the hosts. In Cloudsched, it is defined as the maximum number of loads on the hosts.
3. Metrics on energy efficiency
   - Model: Energy consumption model depends on the disk storage, computation, processing and datacentre cooling system.

2.8 SUMMARY

This chapter reviewed all the major state of the art works in the area of prediction of cloud workloads and optimization of different quality of service parameters. The observation from the relevant work are summarized in the Table 2.1 to 2.6. The chapter concluded with an outcome of the literature review. The next chapter discuses about the conceptual framework designed to overcome the existing prediction and optimization problem.

Chapter 3

CONCEPTUAL FRAMEWORK FOR WORKLOAD MANAGEMENT

Conceptual frameworks are related to concepts, empirical analysis and critical knowledge systematisation theories. The conceptual framework aims to make research findings more meaningful and acceptable to the theoretical perspectives in the field of research.

In this thesis the framework is based on the concepts which constitute the key variables during the analysis. It consists of definitions interconnected to illustrate their relationships.

3.1 WORKLOAD MANAGEMENT FRAMEWORK

The objective of the proposed framework is that (1) the conceptual components are grouped into a framework and (2) dependencies are defined between each module. The end-user may move the workload to the cloud using this system unless it is possible to process it with in-house computing resources. The Figure 3.1 demonstrates the connection between various modules used in cloud workload management.

In general, cloud computing processes workloads such as transactional, batch, analytic, high-performance workloads. From the Providers viewpoint, due to the number of requests and different kinds of workloads, it is very difficult to schedule all workload requests in an effective way. Cloud workload allocation is therefore becoming a tedious task. In addition to allocation, optimization also plays a major role in delivering multiple QoS constraints to cloud resources. Using metaheuristic methods, with a good amount of time, it is proven to get the nearest optimal solution.

- **Workload generator**: We have defined workloads in this thesis as a number of

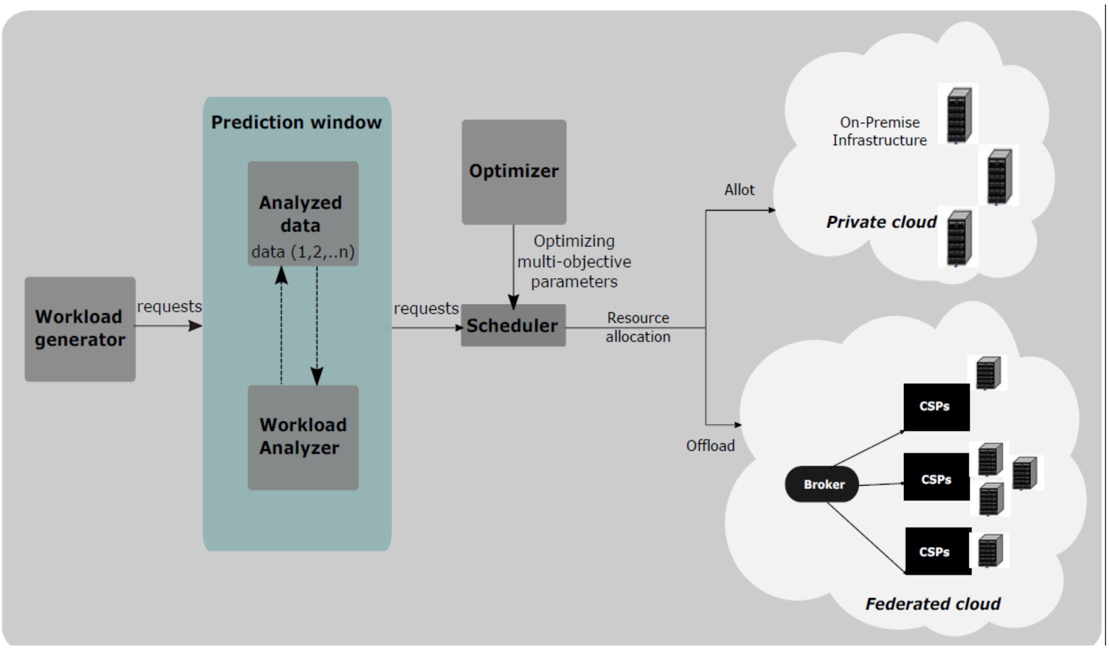


Figure 3.1 Conceptual framework for workload management

jobs that are subdivided into a number of tasks. Workloads may be real or synthetic forms, where real workloads are observed on a broader operating system, and synthetic workloads can be repeatedly implemented in a controlled manner and real workloads can be imitated. The workloads are fed into the prediction window for analysing and predicting.

- **Prediction window**: The load that is generated is given to the prediction window to predict future data. Extraction methods aid in analysing the data in this module. Therefore using prediction algorithms, future data is predicted based on the incoming data.

- **Optimizer**: To effectively distribute cloud resources, it is important to optimise some parameter efficiency.
  The optimization module therefore helps to configure the service parameters to effectively schedule the services to the federated environment which includes optimization techniques.

The prototype thus shows the relationship between the components, which are interdependent on each other.

The Predictor module analyzes and forecasts incoming workloads based on historical data. The resources are either allocated or auto-scaled based on this data. Metaheuristic methods can be used to optimise the parameters to achieve an optimal solution.

This is useful when there are no adequate computational resources for the end user to process the workloads. If it is not possible with in-house computing resources, broker can use this to release the workloads to the federated cloud.

3.2 CLOUD DATA CENTRE ARCHITECTURE

Data centre architecture is presented in Figure 3.2. The client requests are provided to the resource manager along with the predicted output.

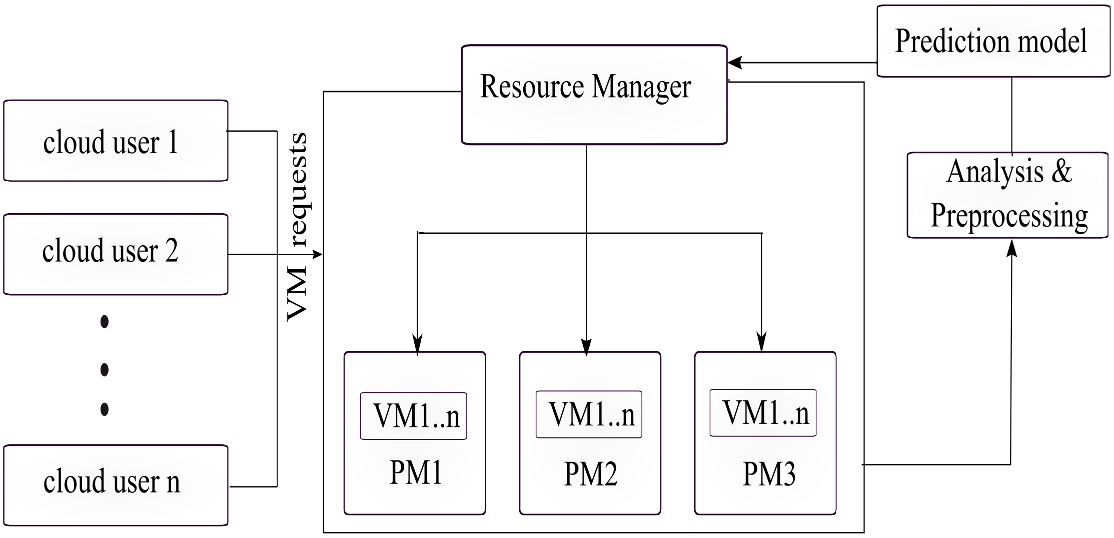


Figure 3.2 Architectural design of Cloud Datacentre

The components involved in the architecture are as follows:

- **Cloud users**: The users or the customers requests the service by paying for the particular service. The requests are forwarded to a resource manager:
- **Resource manager**: Helps in managing all the requests from the cloud users. Resource manager serves the user requests by providing them with a virtual machine.
- **Analysing and pre-processing model**: The workloads are analysed here by applying various filtering techniques and extracting the relevant features. The attribute is pre-processed for future predictions.
- **Prediction model**: Prediction algorithm/model helps in predicting the future demands by analysing the historical data. Various techniques such as statistical, learning methods are useful in predicting the future workloads.

The overall workflow is as follows; the user sends the request to the cloud, where the service broker helps to locate the best cloud service provider. The resulting output from the prediction model is given to the resource manager consisting of the n number of virtual machines, who can then decide on resource autoscaling. The cloud broker, which is the gateway between the user and the provider helps to find the right service provider. The scheduler assists in assigning the loads to on-site facilities or discharging the data to the federated cloud datacenter. The prediction outcome allows the scheduler to have advanced knowledge of the incoming workloads and the services that need to be extended. And, thus, helps in autoscaling the resources.

3.3 BEEM-NN: BEE MUTATION NEURAL NETWORK ARCHITECTURE

The novel BeeM-NN framework is shown in Figure 3.3. Proposed model consists of three phases namely, Feature extraction, Workload prediction and Optimization approach.

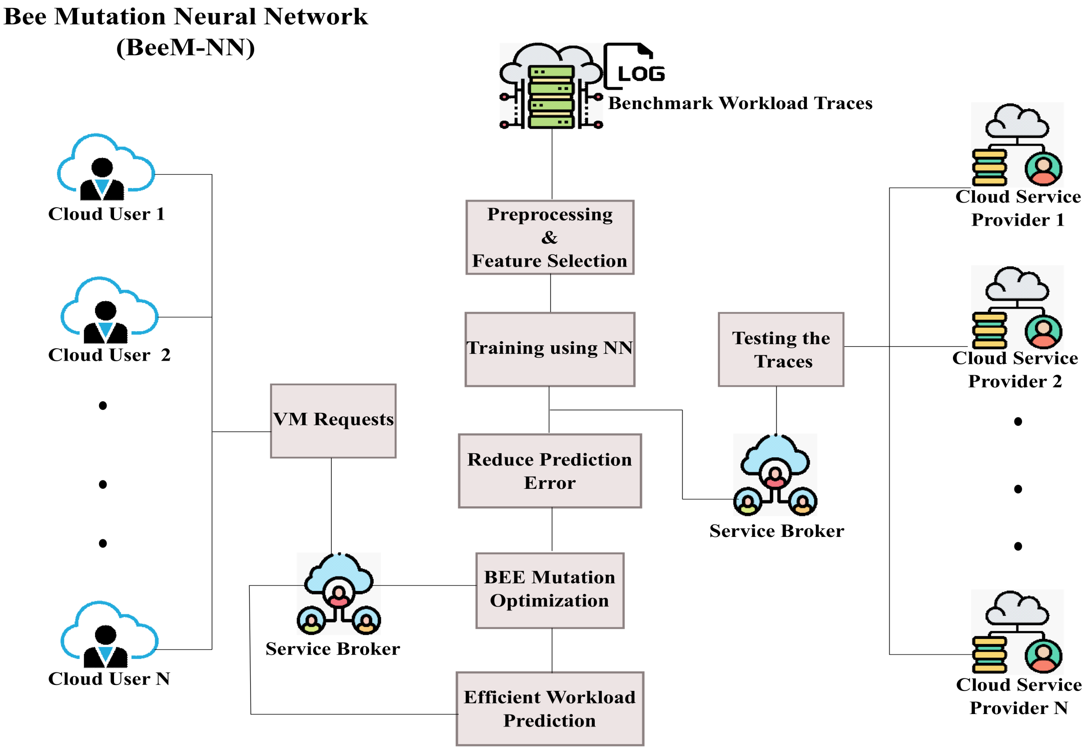


Figure 3.3 Bee Mutation Neural Network (BeeM-NN)

BeeM-NN architecture consists of Prediction and optimization techniques. Prediction technique involves Neural network for processing and optimization methods includes bio inspired hybrid techniques. Following components are involved in the architecture:

- **Cloud users**: In the thesis, we consider the raw inputs i.e., the real workloads from the Microsoft Azure traces which includes Virtual Machines ID, CPU utilization, Memory, Disk space.

- **Cloud Service Provider**: The cloud provider provides the services. In the thesis we have considered Federated cloud service providers.

- **Service Broker**: Service Broker helps in serving the users request by finding the best cloud service provider to satisfy the particular request.

- **Pre-processing and Feature selection**: the workload is analysed and pre-processed to get the Features set. The selected features are used a training set for the Neural Network.

- **Bee Mutation Optimization**: Hybridization of Artificial Bee Colony and Genetic mutation are included here.

The workflow is as follows:

The raw data from Microsoft is collected and further used for pre-processing. The work mainly focuses on the broker level. In the pre-processing step, we have used fitness feature extraction technique to select the best features. We have also performed statistical testing to check the effectiveness of attributes and their dependency on each other. The best features are selected and used as a training set. Training set is used to train the Proposed neural network model. 20% of the traces are used for the test datasets. Modified activation function is used to reduce the prediction error with better prediction accuracy. Next step involves Optimization of Quality of service parameters. Overall In this thesis work, we focus on feature extraction & effective prediction of future workloads, and optimizing the service parameters for effective management of resources.

3.4 ABSTRACT DESIGN OF A PREDICTION WINDOW

Figure 3.4 shows the abstract design of the Prediction module. Historical workload information is analysed and fed into the load predictor, which outputs the future demands and the estimated number of resources. Thus, when a user sends the request also called workloads, Application Provisioner will communicate with an incoming estimated number of resources from the predictor and requests for the scheduling of resources.

- **Workload analyzer**: User historical data is processed, analyzed and fed into the load predictor.
- **Admission control**: User submitted task is processed and sent to Provisioner for further process.
- **Load Predictor**: Outputs the future demands and the estimated number of resources.
- **Application Provisioner**: When a user sends the request also called workloads, the Application Provisioner will communicate with an incoming estimated number of resources from the predictor and requests for the scheduling of resources.

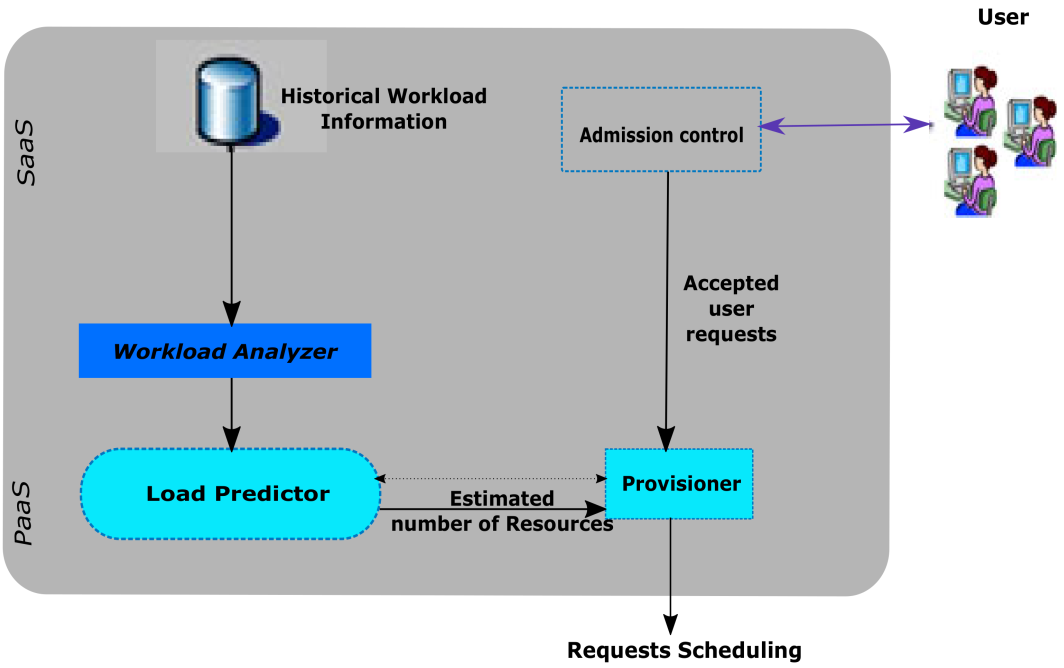


Figure 3.4 Abstract design of Prediction module

Workload prediction problem is resolved effectively by including deep learning techniques. Having a historical workload in the database for a specific time periods, one can efficiently predict the incoming workloads. Machine learning helps in learning the

patterns and extract the features to make the decisions. Historical workload data is considered in a training window which is used to predict the future workloads. Testing data is included in a prediction window which is used after prediction.
Steps involved in the prediction of workloads.
Step 1: Task: Prediction based on historical workload
Step 2: Input1<-Workload sample
Step 3: Input2<- Historical data
Step 4: Include pattern matching technique and extract features.
Step 5: For extracted feature<-use data mining techniques for efficient prediction.

3.5 PREDICTION APPROACH WORKFLOW

The proposed approach to prediction consists of subsequent phases as shown in the Figure 3.5. The historical raw input traces are extracted and analyzed. Unwanted ambiguities are omitted and fed into the model for feature extraction. Pre-processing is done here. The resulting featured information is loaded into the neural networks, where the data is scaled from 0 to 1. For performance reliability, the final output is evaluated and validated.

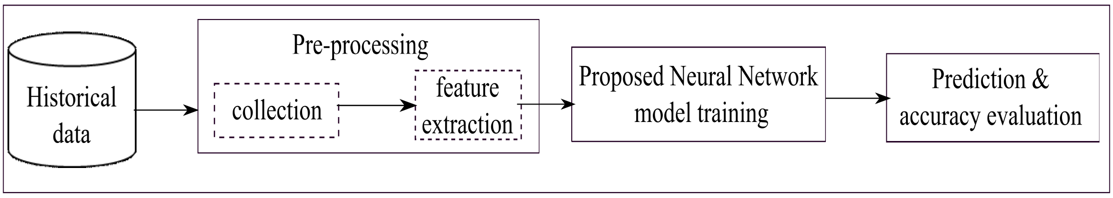

Historical data
Pre-processing
collection
feature extraction
Proposed Neural Network model training
Prediction & accuracy evaluation

Figure 3.5 Prediction approach workflow

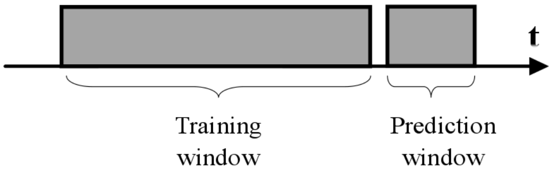


Figure 3.6 Prediction and train window

The data are mined in the training window and used to predict the loads in the prediction window for time t. Figure 3.6 shows the training and testing windows.

3.6 MODULE COMPONENTS

Component diagram (Figure 3.7) is provided in the thesis for easier understanding of the important stages involved in the proposed work.

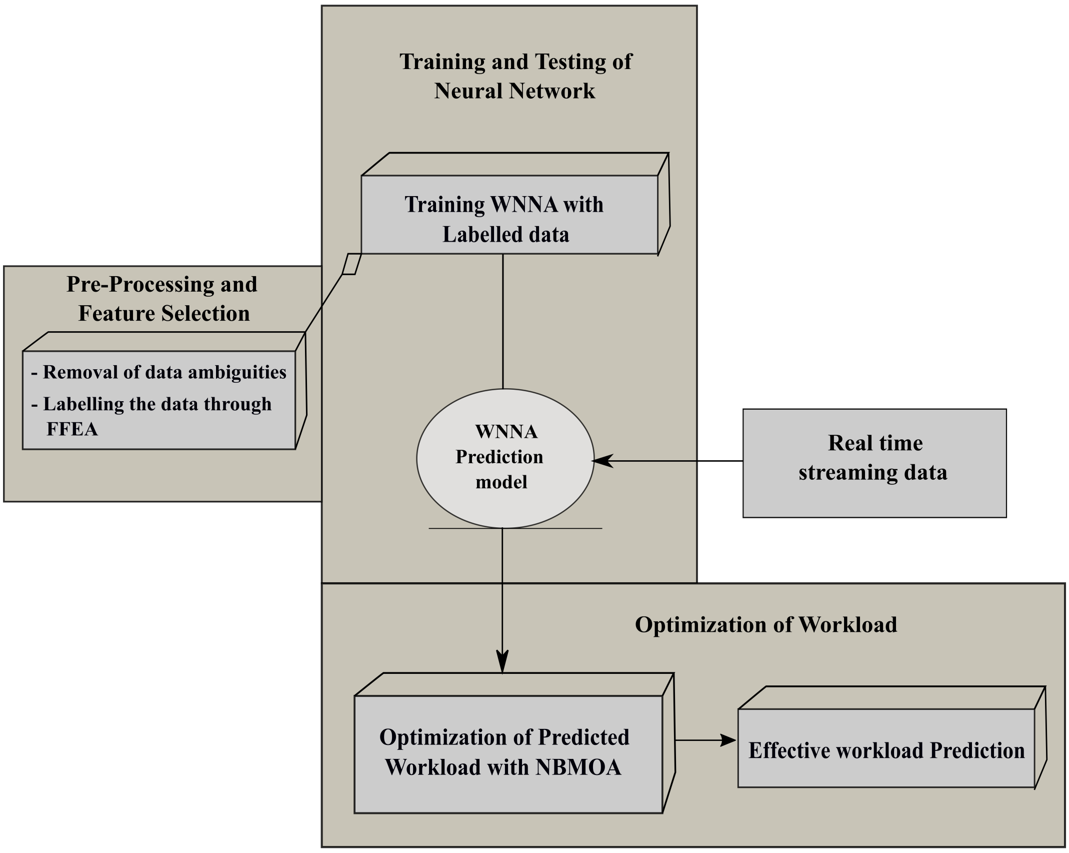

Training and Testing of Neural Network
Training WNNA with Labelled data
Pre-Processing and Feature Selection
- Removal of data ambiguities
- Labelling the data through FFEA
WNNA Prediction model
Real time streaming data
Optimization of Workload
Optimization of Predicted Workload with NBMOA
Effective workload Prediction

Figure 3.7 Conceptual module components

Each module shown in the figure is an independent unit, which is combined to construct the proposed work.

Each module in the system is an autonomous entity which is interdependent with each other. Elimination of data uncertainties and attribute selection is achieved in the pre-processing unit. During the training of the proposed Neural Network model, this corresponding information is used. To validate the model, real time streaming data are obtained. In this unit, training, and testing are performed to get the predicted output. Pre-processing & Feature selection unit detects and removes the data ambiguity and also extracts the features for data labelling. This resultant information is used for training the proposed Neural network model. Training and testing are done in this unit to get the predicted output. Prediction workload is optimized using the proposed hybrid model to effectively optimize different quality parameters. Each module is explained clearly in the subsequent chapters.

3.7 SUMMARY

A conceptual framework is designed in this chapter consists of numerous concepts that are interlinked with each other. Each module is given a brief description and this chapter presents the proposed system architecture. In the following chapters, subsequent module experiment results are shown.

Chapter 4

PROPOSED FITNESS FEATURE EXTRACTION MODEL

The set up for workload is established with several components where several cloud users place the new VM request for resource allocation. The VM request is processed through the service broker, and the requested resource is allocated effectively. This process of computing the required resource and its allocation increase the workload in the cloud environment.

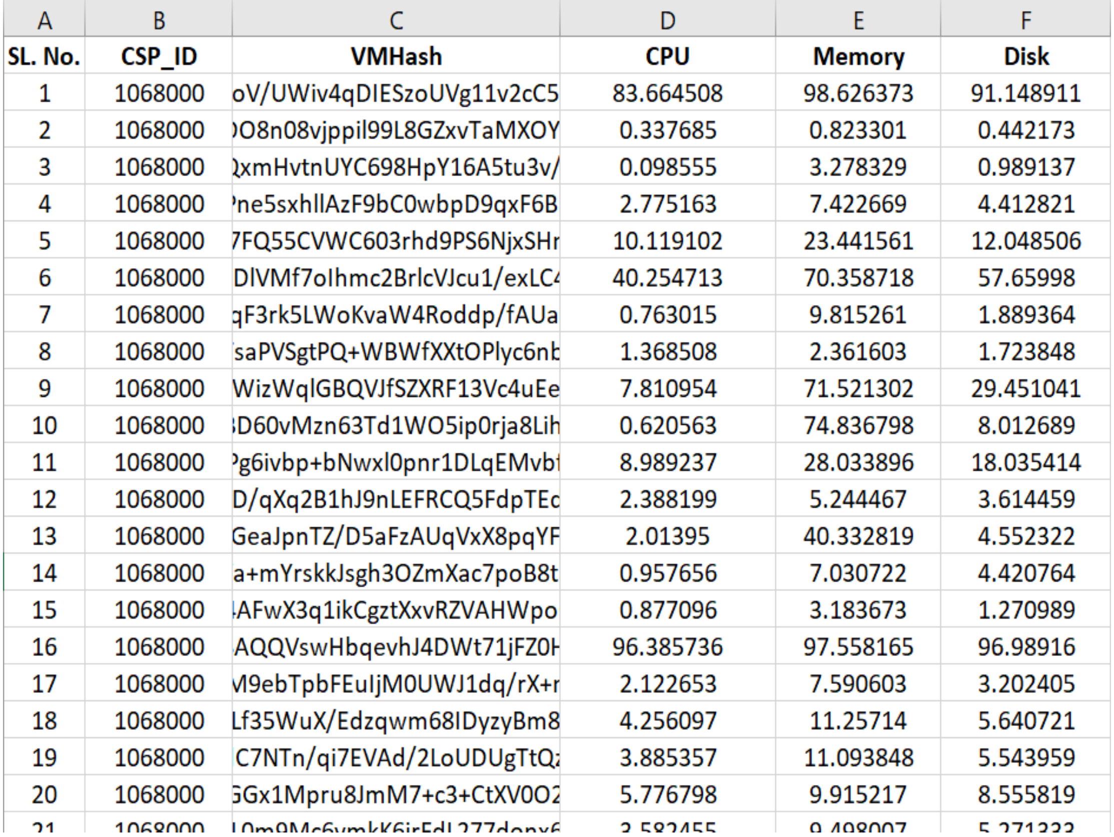
Three significant factors are considered namely, CPU, Disk and Memory which plays a major role in workload prediction.

4.1 DATA COLLECTION

The workload traces are collected from Azure which is publicly available. Workload traces obtained from Azure Public Dataset (http://azurepublicdataset.blob.core.windows.net/trace_data/vm_cpu_readings-file-52-of-125.csv.gz) (Microsoft, 2019). The collected dataset are pre-processed initially to remove the noises. Figure 4.1 shows the sample Azure public dataset. Since the dataset employed in the model consists of a vast volume of workload traces, it is treated with the practical dimensional reduction approach for both pre-processing and selection.

The data traces contain service providers Identification numbers and VM hash data. Along with it, the traces contain CPU values, Memory values and Disk values.

- CPU: The amount of processor being used.
- Memory: The amount of RAM being used
- Disk: the amount of storage (HDD, SSD) being used.



A	B	C	D	E	F
SL. No.	CSP_ID	VMHash	CPU	Memory	Disk
1	1068000	oV/UWiv4qDIESzoUVg11v2cC5	83.664508	98.626373	91.148911
2	1068000	)O8n08vjppil99L8GZxvTaMXOY	0.337685	0.823301	0.442173
3	1068000	QxmHvtnUYC698HpY16A5tu3v/	0.098555	3.278329	0.989137
4	1068000	'ne5sxhllAzF9bC0wbpD9qxF6B	2.775163	7.422669	4.412821
5	1068000	7FQ55CVWC603rhd9PS6NjxSHr	10.119102	23.441561	12.048506
6	1068000	DlVMf7oIhmc2BrlcVJcu1/exLC4	40.254713	70.358718	57.65998
7	1068000	qF3rk5LWoKvaW4Roddp/fAUa	0.763015	9.815261	1.889364
8	1068000	'saPVSgtPQ+WBWfXXtOPlyc6nb	1.368508	2.361603	1.723848
9	1068000	WizWqlGBQVJfSZXRF13Vc4uEe	7.810954	71.521302	29.451041
10	1068000	3D60vMzn63Td1WO5ip0rja8Lih	0.620563	74.836798	8.012689
11	1068000	'g6ivbp+bNwxl0pnr1DLqEMvbf	8.989237	28.033896	18.035414
12	1068000	D/qXq2B1hJ9nLEFRCQ5FdpTEc	2.388199	5.244467	3.614459
13	1068000	GeaJpnTZ/D5aFzAUqVxX8pqYF	2.01395	40.332819	4.552322
14	1068000	a+mYrskkJsgh3OZmXac7poB8t	0.957656	7.030722	4.420764
15	1068000	4AFwX3q1ikCgztXxvRZVAHWpo	0.877096	3.183673	1.270989
16	1068000	AQQVswHbqevhJ4DWt71jFZ0H	96.385736	97.558165	96.98916
17	1068000	M9ebTpbFEuIjM0UWJ1dq/rX+r	2.122653	7.590603	3.202405
18	1068000	Lf35WuX/Edzqwm68IDyzyBm8	4.256097	11.25714	5.640721
19	1068000	C7NTn/qi7EVAd/2LoUDUgTtQz	3.885357	11.093848	5.543959
20	1068000	GGx1Mpru8JmM7+c3+CtXV0O2	5.776798	9.915217	8.555819
21	1068000	L0m9Mc6vmkK6irFdL277donx6	3.582455	9.498007	5.271333

Figure 4.1 Azure public data traces

4.2 FITNESS FUNCTION MODEL

The Fitness function algorithm estimates the feature function and verifies whether the usage variable of CPU, memory and disk values fall within the fitness limits.

Algorithm 1 Fitness Function

```
Procedure Label = Fitness(Var, X)
Label = -1
if $(0 \leq Var \leq X(1))$ then Label = 0
else if $Var > X(end)$ then Label = 6
else
    for $i = 1 to |X|$ do
        if $X(i) < Var \leq X(i+1)$ then Label = i break
        end if
    end for
end if
```

When the value is less than the minimum fitness limits, the corresponding label is assigned to 0 (No load) and more than the maximum fitness value, yields label to value 6 (Shut down) which is described in Algorithm 1. The average costs are determined based on the Fitness values given in Table 4.1.

Table 4.1 Fitness values with labels for three inputs

Labels	**CPU**	**MEMORY**	**I/O**
No Load	0-5	0-8	0-8
Light Load	6-15	9-18	9-20
Gentle Load	16-35	19-35	21-40
Medium Load	36-60	36-65	41-70
Strong Load	61-90	66-90	71-90
Extreme Load	91-100	91-100	91-100
Shut down	More than 100	More than 100	More than 100

Table 4.1 provides the range values for corresponding CPU, memory and Disk space. More than 100% signifies that the Systems shutdown and tasks are abandoned.

4.3 PRE-PROCESSING AND FEATURE SELECTION

The feature extraction for the proposed model is obtained through the Fitness Feature Extraction Algorithm (FFEA) (Algorithm 2). The obtained values of CPU, disk and memory are evaluated with the fitness values, and the corresponding labelling is performed.

Based on the obtained values, final labels are classified as No-load, Light load, Gentle load, Medium load, Strong load, Extreme load and Shutdown for values from 0 to 6 as shown in Figure 4.2. From the final label, the class label is generated in the form of Allow and Denied for 0 to 6 values respectively.

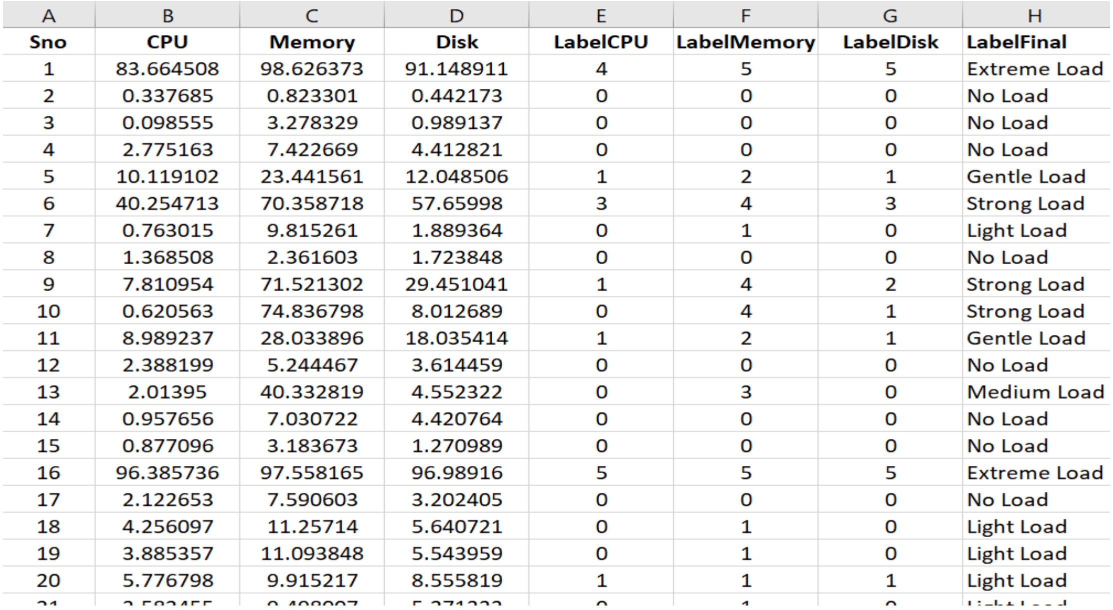
4.3.1 Pre-processing workflow

Figure 4.3 provides a summary of the approach presented. It consists of six layers.

Algorithm 2 Fitness Feature Extraction Algorithm

```
Procedure [LabCpu, LabMemory, LabDisk, LabFinal, LabClass]= FFEA(Cpu, Memory, Disk)
A = [5 15 35 60 90 100]
B = [8 18 35 60 90 100]
C = [8 20 40 60 90 100]
for i=1 to N do
    LabCpu(i) = Fitness(Cpu(i), A)
    LabMemory(i) = Fitness(Memory(i), B)
    LabDisk(i) = Fitness(Disk(i), C)
    if LabCpu > LabMemory && LabCpu > LabDisk then
        LF = LabCpu
    else
        if LabMemory > LabDisk then
            LF = LabMemory
        else
            LF = LabDisk
        end if
        if LF == 0 then LF = “No Load”
            LC = “Allow”
        else if LF == 1 then LF = “Light Load”
            LC = “Allow”
        else if LF == 2 then LF = “Gentle Load”
            LC = “Allow”
        else if LF == 3 then LF = “Medium Load”
            LC = “Allow”
        else if LF == 4 then LF = “Strong Load”
            LC = “Denied”
        else if LF == 5 then LF = “Extreme Load”
            LC = “Denied”
        else if LF == 6 then LF = “Shutdown”
            LC = “Denied”
        end if
        LabFinal(i) = LF; LabClass(i) = LC
    end if
end for
```

In Step 1, historical data is obtained and analyzed. Domain-specific important features are extracted, & ambiguities are removed from the raw data in Step 2 and 3. Additional Features are extracted in Step 4. In step 5, The data is divided into training and testing sets. Results are obtained from these sets for better prediction accuracy in Step 6. Each element is discussed in detail in the next section.



A	B	C	D	E	F	G	H
Sno	**CPU**	**Memory**	**Disk**	**LabelCPU**	**LabelMemory**	**LabelDisk**	**LabelFinal**
1	83.664508	98.626373	91.148911	4	5	5	Extreme Load
2	0.337685	0.823301	0.442173	0	0	0	No Load
3	0.098555	3.278329	0.989137	0	0	0	No Load
4	2.775163	7.422669	4.412821	0	0	0	No Load
5	10.119102	23.441561	12.048506	1	2	1	Gentle Load
6	40.254713	70.358718	57.65998	3	4	3	Strong Load
7	0.763015	9.815261	1.889364	0	1	0	Light Load
8	1.368508	2.361603	1.723848	0	0	0	No Load
9	7.810954	71.521302	29.451041	1	4	2	Strong Load
10	0.620563	74.836798	8.012689	0	4	1	Strong Load
11	8.989237	28.033896	18.035414	1	2	1	Gentle Load
12	2.388199	5.244467	3.614459	0	0	0	No Load
13	2.01395	40.332819	4.552322	0	3	0	Medium Load
14	0.957656	7.030722	4.420764	0	0	0	No Load
15	0.877096	3.183673	1.270989	0	0	0	No Load
16	96.385736	97.558165	96.98916	5	5	5	Extreme Load
17	2.122653	7.590603	3.202405	0	0	0	No Load
18	4.256097	11.25714	5.640721	0	1	0	Light Load
19	3.885357	11.093848	5.543959	0	1	0	Light Load
20	5.776798	9.915217	8.555819	1	1	1	Light Load

Figure 4.2 Training datasets after FFEA

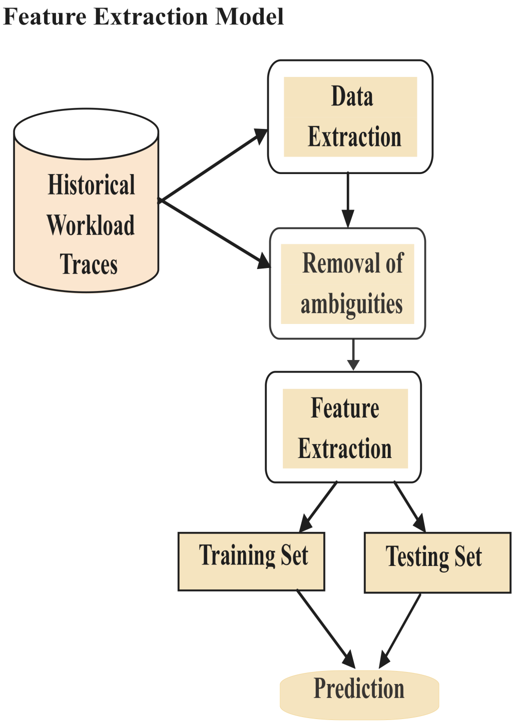


Figure 4.3 Pre-processing flowchart

4.4 SUMMARY

In this chapter, Data pre-processing workflow is discussed. Data has been collected from Microsoft Azure, and features are selcted for processing. Three important attributes such CPU, Memory and Disk space are extracted and analyzed using the proposed Feature extraction model. Raw datasets along with the training datasets are represented and Pre-processing workflow has been provided.

Chapter 5

NEURAL NETWORK ALGORITHM FOR WORKLOAD PREDICTION

This chapter discusses the proposed Workload Neural Network Algorithm (WNNA) and corresponding performance analysis of the testing and training data.

5.1 PRELIMINARIES

5.1.1 Recurrent Neural Network

A simple Recurrent Neural Network (RNN) comprises up to three critical layers, one for the feature input and other its output. The processing is performed in the hidden layer. Generally, the values from the dataset *i* provided to the RNN defines the range of input layers.

The input layers are provided with the input vectors at sequential time t. The hidden layer units in the network are connected to the input layer units through the computed weight values $W_{I \to H}$. All the hidden layer units are linked to one another through the time function (Salehinejad et al., 2017). The hidden layer is defined with the activation function A_{fi} which is given in Eq (5.1) as,

$$h(t) = A_{fi}(c_t), \tag{5.1}$$

where

$$c_t = W_{I \to H} i_t + W_{H \to H} h_{t-1} + b_h \tag{5.2}$$

In Eq (5.2), b_h is the hidden unit vector bias. The hidden units are then connected to the output layer with the weight of $W_{H \to O}$. Similar to the connection between input and hidden layer, the hidden-output layer connection also has activation function A_{fo} and its corresponding bias vector b_o. The output layer is defined in Eq (5.3) as

$$O_t = A_{fo}(W_{H\to O}h(t)+b_o) \tag{5.3}$$

The activation function can be mainly of two types as sigmoid and tanh. They are defined with the input data 'x' in Eq (5.4) and Eq (5.5) as follows

$$f(x) = \frac{1}{1+e^{-x}} \tag{5.4}$$

and

$$\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1} \tag{5.5}$$

5.2 WORKLOAD NEURAL NETWORK TRAINING PHASE

Figure 5.1 shows the framework of WNNA in predicting the workload. The selected features are extracted and labeled through FFEA and fed into RNN scheme with the novel activation function named as Workload Neural Network (WNN).

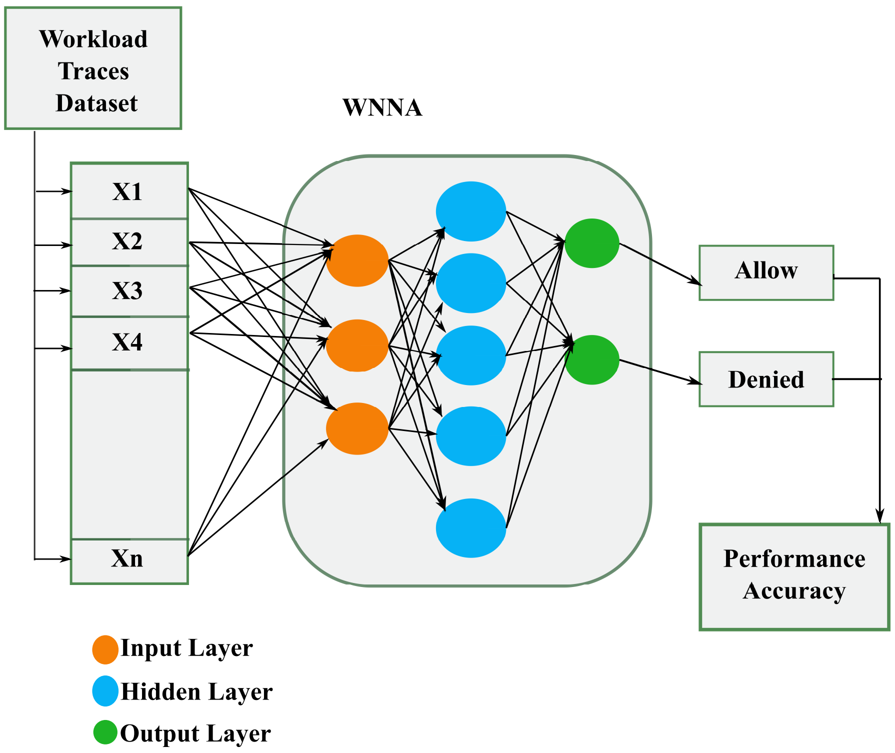


Figure 5.1 Framework of prediction module

Algorithm 3 Workload Neural Network Algorithm

```
Procedure h = WNNA(x) x = $[x_1, x_2, \ldots, x_n], x_t \in R^m$
Input Parameters = $W_f, U_f, b_f, W_C, U_C, b_C, W_i, U_i, b_i, W_o, U_o, b_o$
$h_0 = Zeroes(1,p); C_0 = h_0$
for t = 1 to n do
    $f(t) = \sigma(W_{fx}(t) + U_{fh}(t-1) + b_f)$
    $C(t) = tanh(W_C(t) + U_C h(t-1) + b_C)$
    $C(t) = 1 - \tanh(C(t)xC(t))$
    $i(t) = \sigma(W_i x(t) + U_i h(t-1) + b_i)$
    $C(t) = f(t) \bigodot C(t-1) + i(t) \bigodot C(t)$
    $O(t) = \sigma(W_o x(t) + U_o h(t-1) + b_o)$
    $h(t) = tanh(C(t))O(t))$
end for
Set Input units, wnna-units, Output units and optimizer
    for WNNA Network (L)
Normalize the dataset ($D_n$) into values from 0 to 1
choose window size for training (tw) and establish $D_n$
    accordingly
for for n = 1 to BS do
    Train the WNN (L)
end for
Run Predictions using L
Estimate the performance metrics with equation
```

The proposed novel WNNA is initially fed with the CPU, memory space and disk labels. After processing through the hidden layer, the results are provided in the output layer. Algorithm 3 gives the WNNA functionality.

The processing over the extracted feature initiates in the hidden layer through the activation function $WNNA_F$ with a modified tan$h$ function. The weight values of the layers and the input features are provided initially. The setup unit, WNNA unit and output unit are established to define the WNNA network. The extracted features from the dataset are normalized to a range of 0 to 1. Finally, the training is performed on the established WNNA for predicting the model generation.

5.3 EXPERIMENTAL SETUP

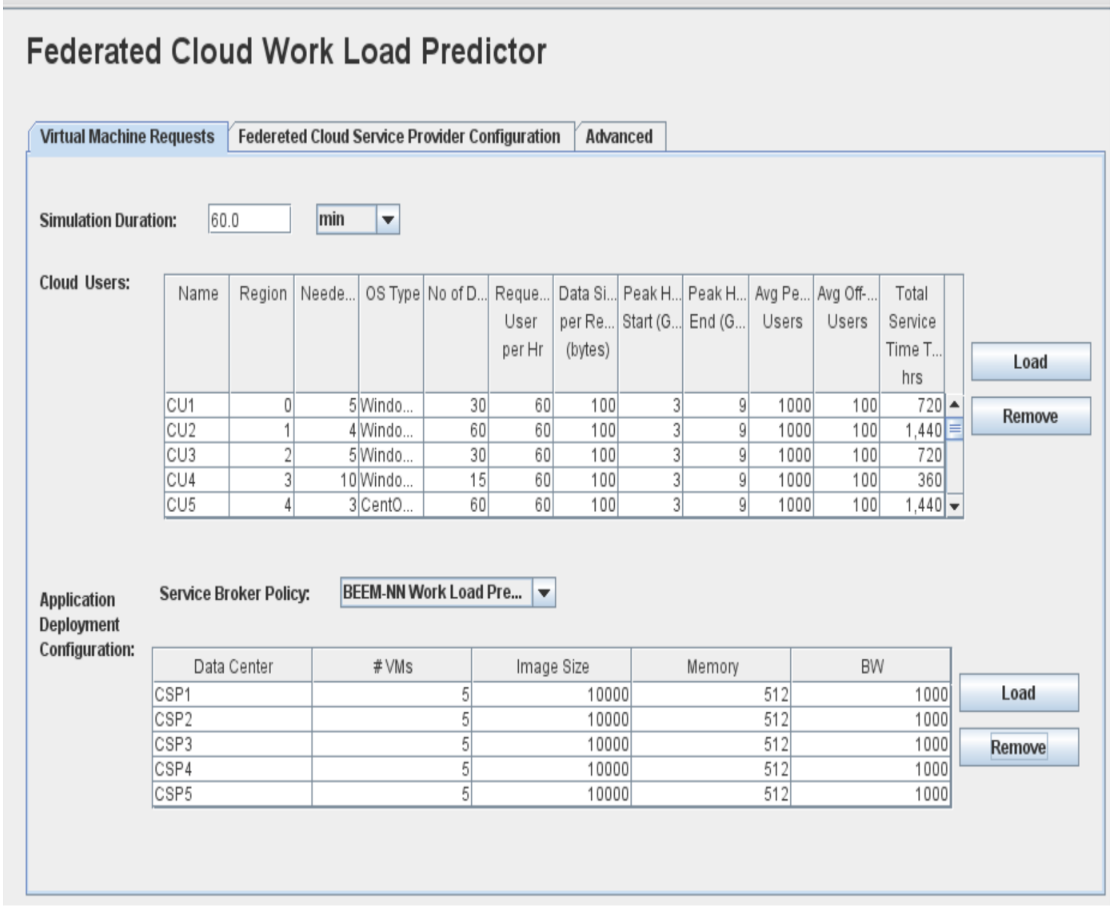
The configuration setup is designed considering many aspects including data centres, geographical regions, users requesting resources. The experiment is done using CloudSim Simulation tool. The service broker handles the VM request and efficiently allocates the services needed. The workload in the cloud environment is increased by this process of assessing and providing the resources needed.

Data is divided into two separate components in scenario one, named training and testing sets. Training data, where the input is used to train the algorithm while the test content is loaded to the evaluation model. 80% of the data is used for training and 20% for testing. In the second scenario, VM configuration is done to generate the test traces i.e., the synthetic workload is generated. Combination of both the test trace scenarios are used as a testing set for the proposed model. Figure 5.2 shows the experimental window in the cloudsim tool for generating the test sets.

The providers of cloud resources in different geographical regions are shown in Figure 5.3. Table 5.1 provides configuration parameters for simulation setup.

Table 5.1 Simulation Configuration

Configuration parameters	
Virtual Machines	35
Geographical regions	6
Datacentre providers	5
VM present in Datacentres	5
Requests per hour	60
Physical hardware unit	1
Simulator	Cloudsim


Federated Cloud Work Load Predictor
Virtual Machine Requests
Federeted Cloud Service Provider Configuration
Advanced
Load
Remove
Load
Remove

Simulation Duration: 60.0 min

Cloud Users:

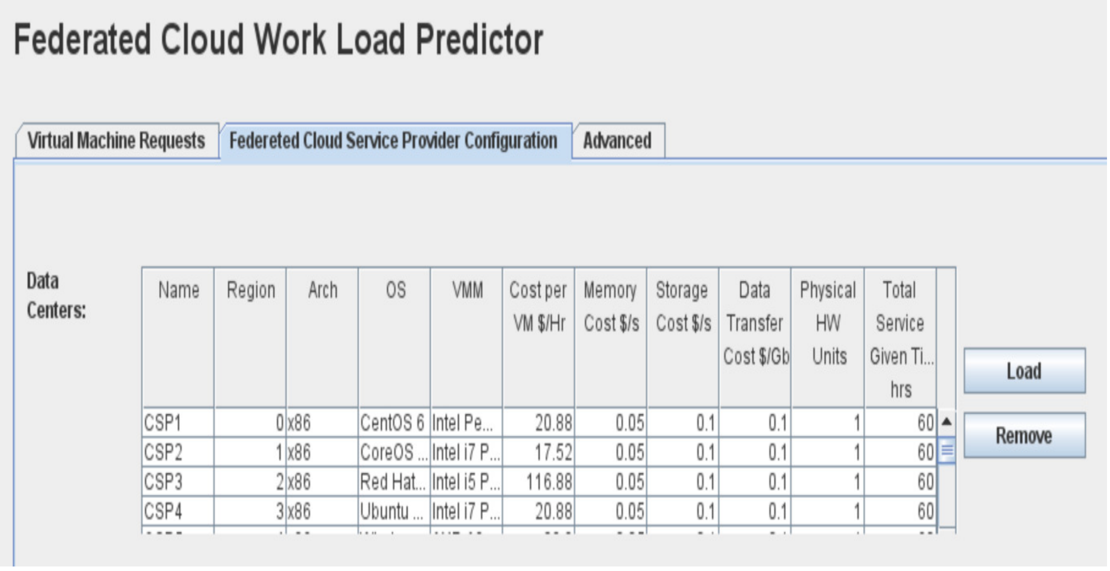
Name	Region	Neede...	OS Type	No of D...	Reque... User per Hr	Data Si... per Re... (bytes)	Peak H... Start (G...	Peak H... End (G...	Avg Pe... Users	Avg Off-... Users	Total Service Time T... hrs
CU1	0	5	Windo...	30	60	100	3	9	1000	100	720
CU2	1	4	Windo...	60	60	100	3	9	1000	100	1,440
CU3	2	5	Windo...	30	60	100	3	9	1000	100	720
CU4	3	10	Windo...	15	60	100	3	9	1000	100	360
CU5	4	3	CentO...	60	60	100	3	9	1000	100	1,440

Application Deployment Configuration:

Service Broker Policy: BEEM-NN Work Load Pre...

Data Center	# VMs	Image Size	Memory	BW
CSP1	5	10000	512	1000
CSP2	5	10000	512	1000
CSP3	5	10000	512	1000
CSP4	5	10000	512	1000
CSP5	5	10000	512	1000

(a) Virtual machine configuration

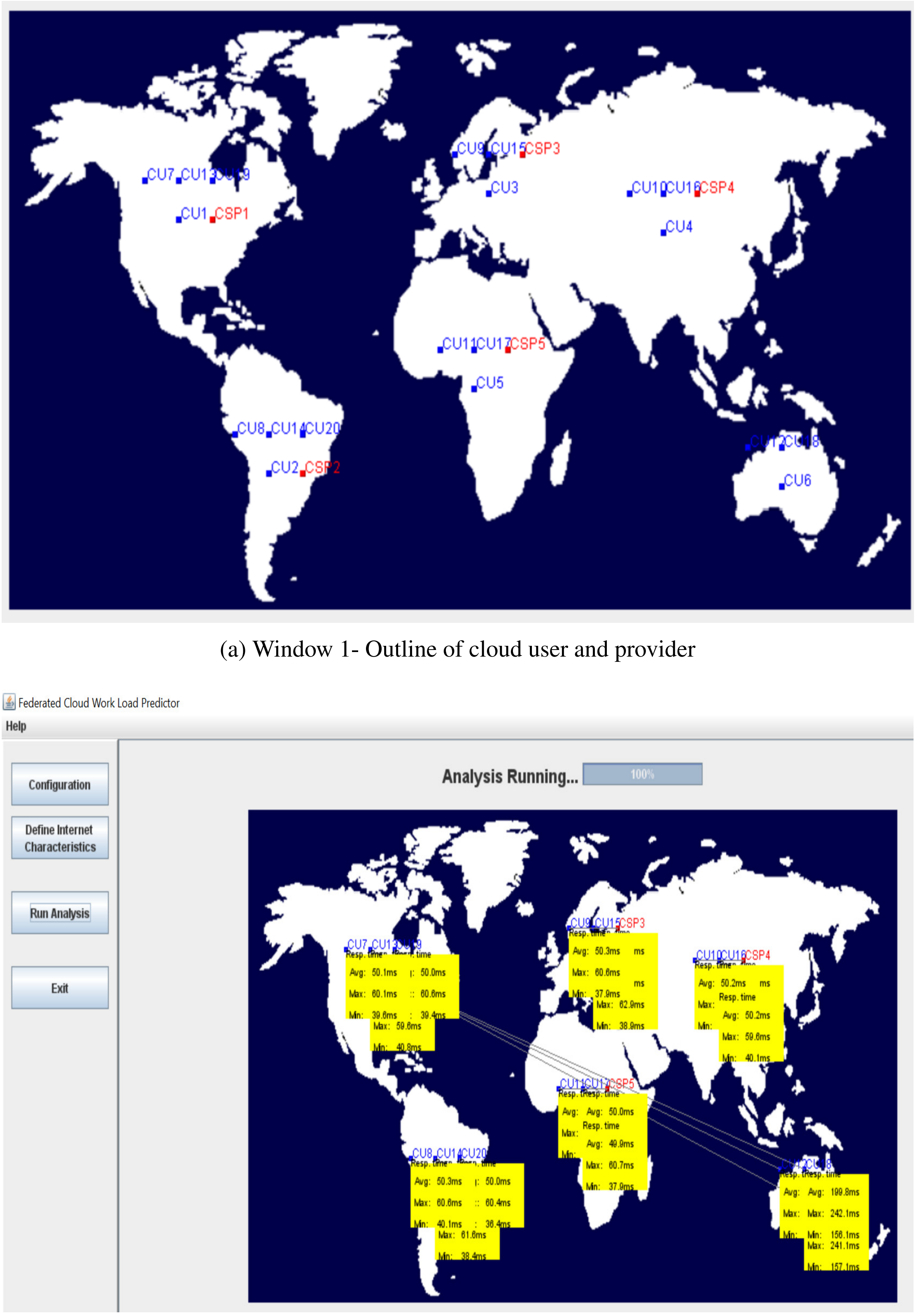

Federated Cloud Work Load Predictor
Virtual Machine Requests
Federeted Cloud Service Provider Configuration
Advanced
Load
Remove

Data Centers:

Name	Region	Arch	OS	VMM	Cost per VM $/Hr	Memory Cost $/s	Storage Cost $/s	Data Transfer Cost $/Gb	Physical HW Units	Total Service Given Ti... hrs
CSP1	0	x86	CentOS 6	Intel Pe...	20.88	0.05	0.1	0.1	1	60
CSP2	1	x86	CoreOS ...	Intel i7 P...	17.52	0.05	0.1	0.1	1	60
CSP3	2	x86	Red Hat...	Intel i5 P...	116.88	0.05	0.1	0.1	1	60
CSP4	3	x86	Ubuntu ...	Intel i7 P...	20.88	0.05	0.1	0.1	1	60

(b) Service provider configuration

Figure 5.2 Simulation window- Federated cloud load predictor.



(a) Window 1- Outline of cloud user and provider

(b) Window 2- Allocation of resources

Figure 5.3 Cloud service providers at different geographical regions.

5.4 DATA ANALYSIS

Figure 5.4 shows the clustered data traces without FFEA (Proposed Fitness Feature Extraction Algorithm).

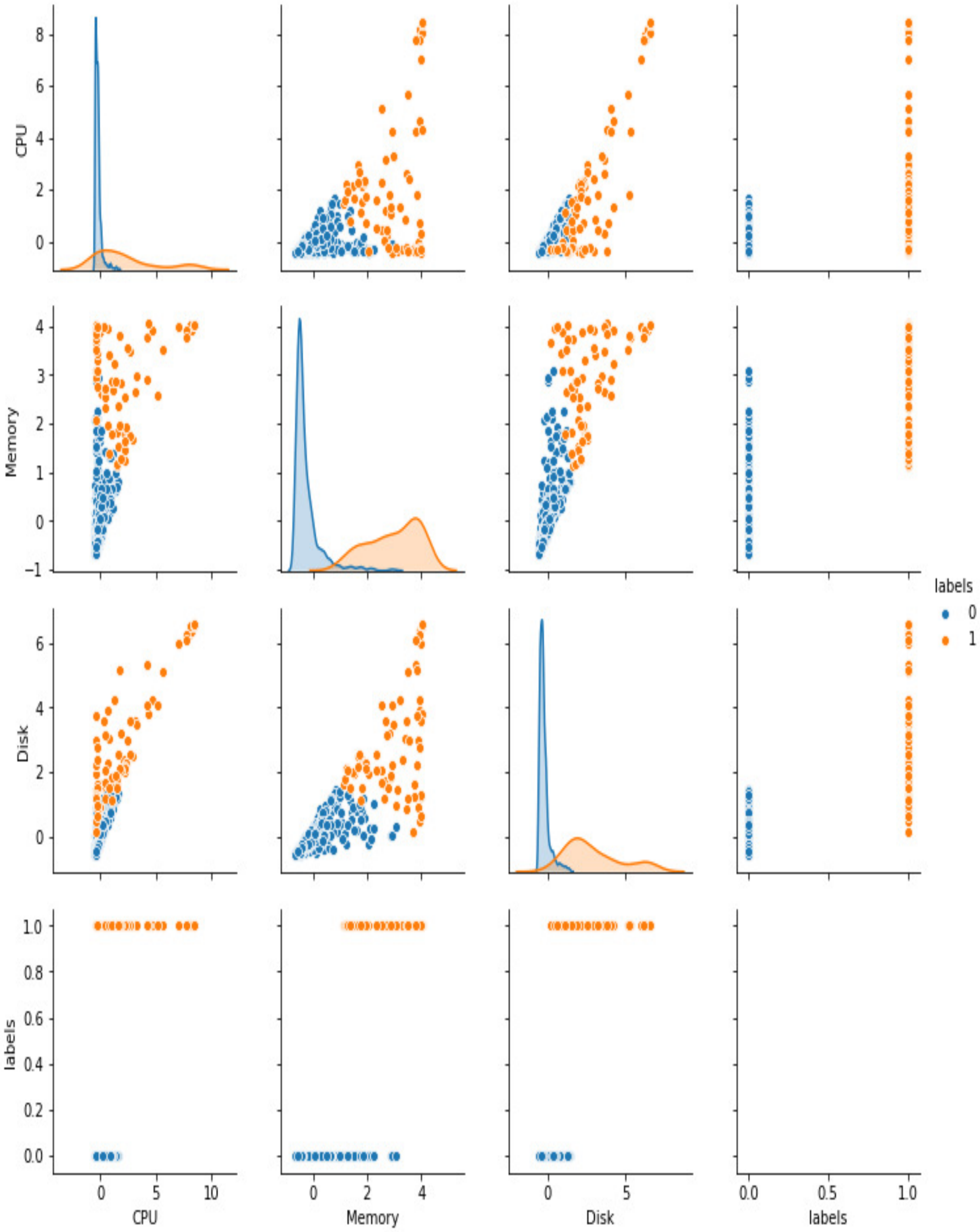


Figure 5.4 Cluster data without FFEA

Initially, k-means clustering is applied on the training data traces. It can be clearly seen from the pairplot graph, that there exist an overlapping of cluster datapoints.

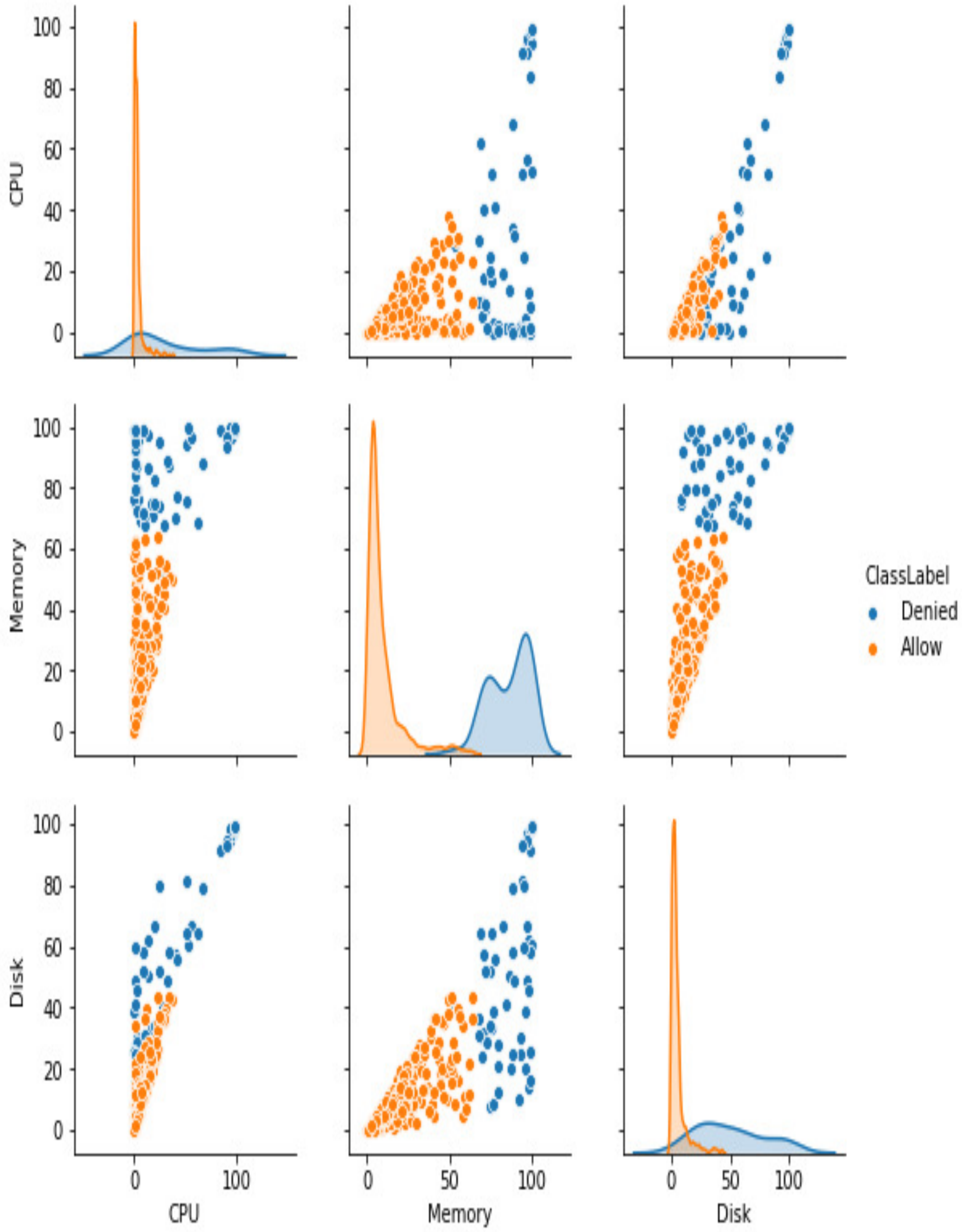


Figure 5.5 Cluster data with FFEA class Labels

Learning process for the model will be difficult if there exists an overlapping between two cluster data points. Classifying the labels into a particular cluster will be difficult. Thus the model is not able to learn the pattern easily to classify the datapoints into a particular cluster. It is observed that, this technique of normal clustering achieved less accuracy.

To improve the accuracy, FFEA is applied. FFEA labelled cluster datapoints is shown in the Figure 5.5, where overlapping has been reduced drastically and datapoints are also correctly classified than the previous clustering method. Applying FFEA and WNNA techniques, we were able to classify the data properly. It is observed that, the proposed model is able to achieve good performance than the current methods.

5.4.1 Statistical Analysis

Statistical hypothesis testing has been performed using T-test method on each input label (CPU, Memory and Disk) corresponding to the output Class Label. The significance level α is chosen as 5% where α = Level of significance = $P(Reject\, H_0 | H_0\, is true)$.
Test statistic t is computed as:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} , \tag{5.6}$$

with

$$s_p = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}} , \tag{5.7}$$

where $\bar{x}_1$ is Mean of CPU allowed label, $\bar{x}_2$ is Mean of CPU denied label. n_1 and n_2 are the number of CPU allowed and denied observation. s_1 and s_2 are the Standard deviation of both allowed and denied labels of CPU input. s_p is the pooled standard deviation which is calculated using equation 5.7.

Table 5.2 Statistical hypothesis testing

Feature	**P-Value**	**Hypothesis**	**Statistical Test Method**	**Significance**
CPU	1.364 e-81	Reject Null	Independent T-Test	Yes
Memory	2.442 e-275	Reject Null	Independent T-Test	Yes
Disk	1.167 e-184	Reject Null	Independent T-Test	Yes

Mean CPU utilization of allowed and denied labels are compared with each other. H_0 signifies the mean CPU utilization of the allowed label. Similarly, comparison is

carried out for both Disk and Memory labels individually. Table 5.2 gives the results obtained by hypothesis testing. Since P value is less than the alpha value, we reject the Null Hypothesis.
This infers, there is a difference in mean value of CPU, memory, disk for allowed and denied labels. Thus, the input labels (CPU, Memory and Disk) strongly impacts the output class label.

5.5 PERFORMANCE ANALYSIS OF TRAIN AND TEST DATA

Performance of the proposed model over training and testing dataset is presented here. Train and test ratio is 80:20. After applying the proposed technique on the training dataset, we have found high accuracy with minimal loss.

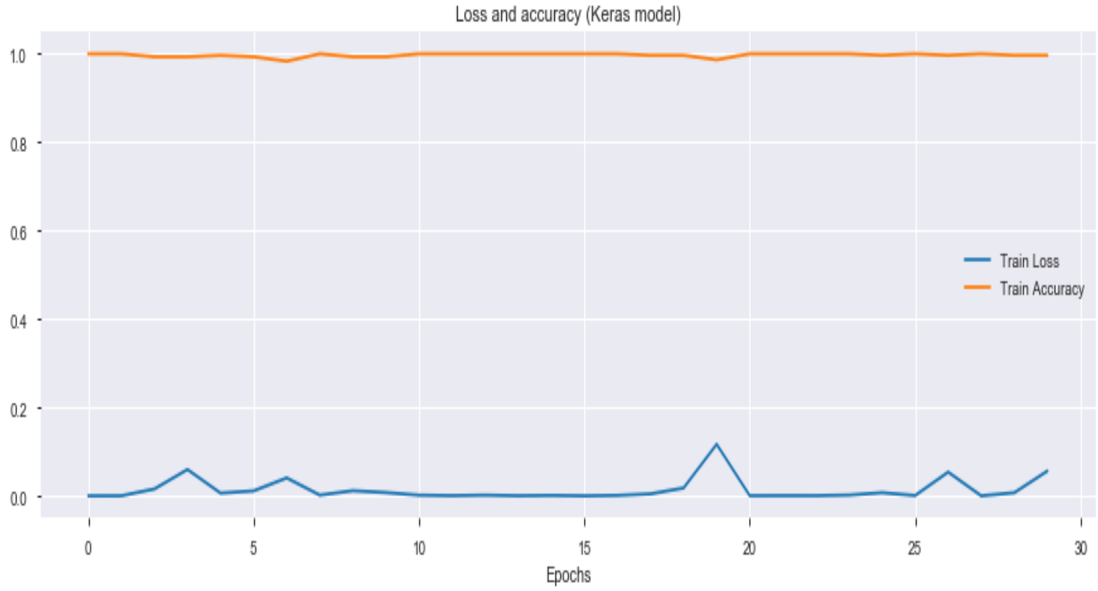


Figure 5.6 Accuracy and Loss of FFEA Training dataset

Figure 5.6 shows the accuracy and loss from the training dataset. Similarly in Figure 5.7, during the testing phase, when the proposed technique is applied on the testing data, it is observed that the model has achieved high accuracy with the minimal error.

At certain epochs, loss has slightly increased due to which accuracy goes down. Training and Testing model accuracy is found to be similar. The proposed model gives good performance on both the training and testing data. Hence, the model is neither an overfit nor an underfit. Enlarged representation of loss and accuracy of Training and

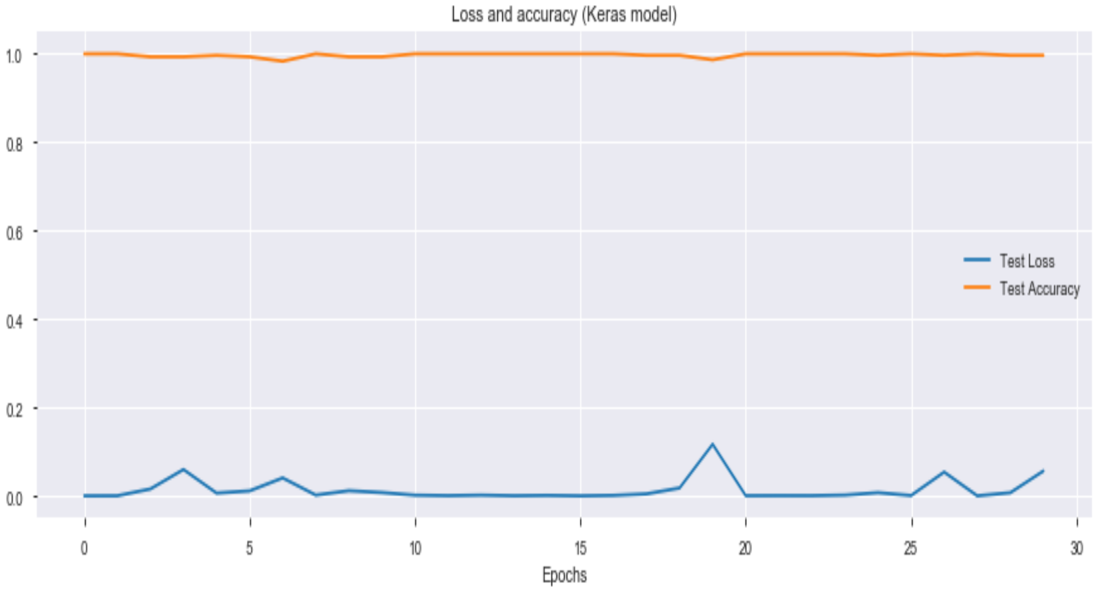


Figure 5.7 Accuracy and Loss of FFEA Testing dataset

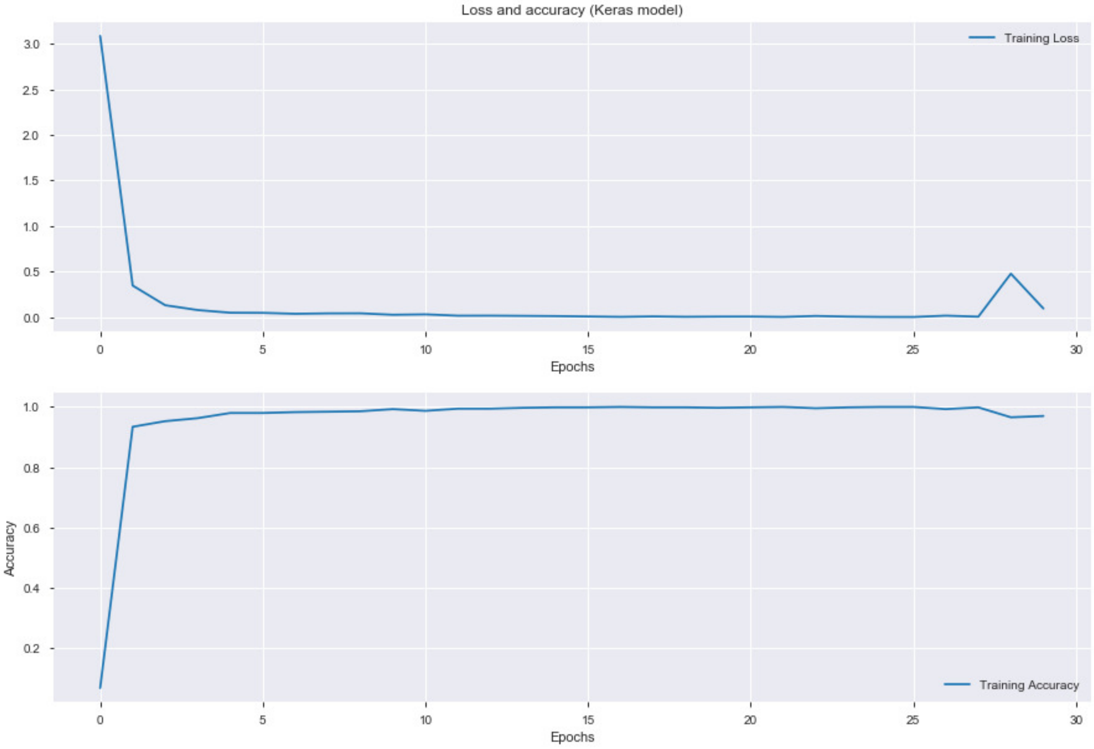


Figure 5.8 Magnified Y-axis for FFEA Training dataset

Testing set are shown in Figures 5.8 and 5.9 respectively.

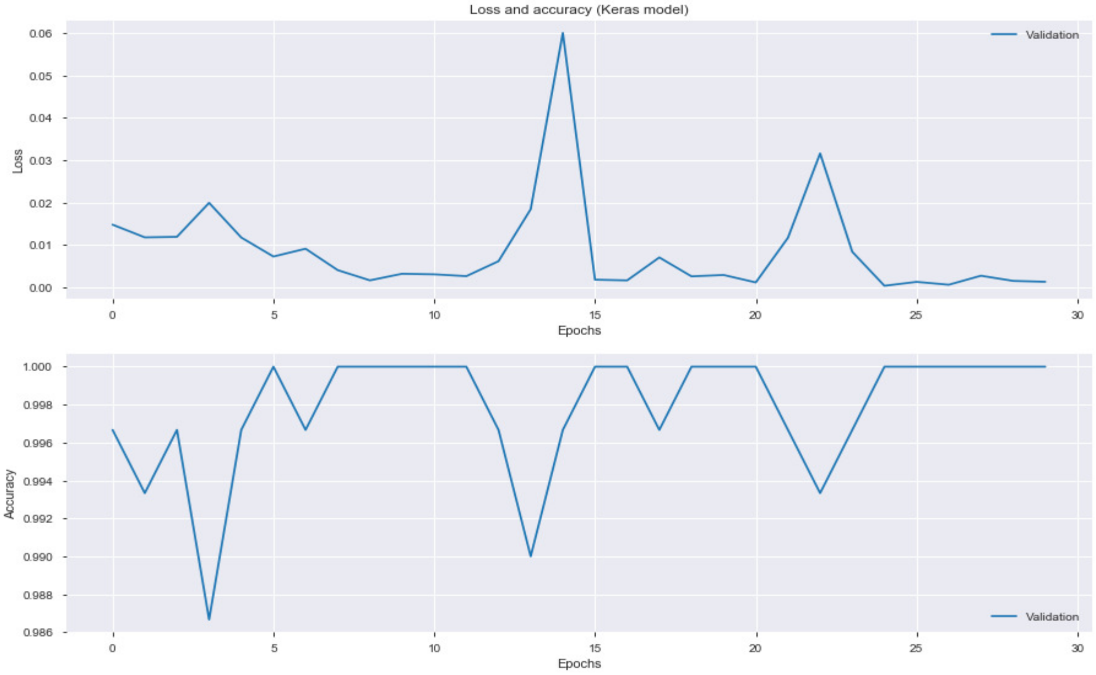


Figure 5.9 Magnified Y-axis for FFEA Testing dataset

5.5.1 Cross-validation

For better performance evaluation of the proposed model, we have also performed k-fold cross validation. In the proposed work, the given data is split into k fold, where k=5. Test and train percentage is set to 20 & 80. The process is repeated until each fold has been used as the testing set.

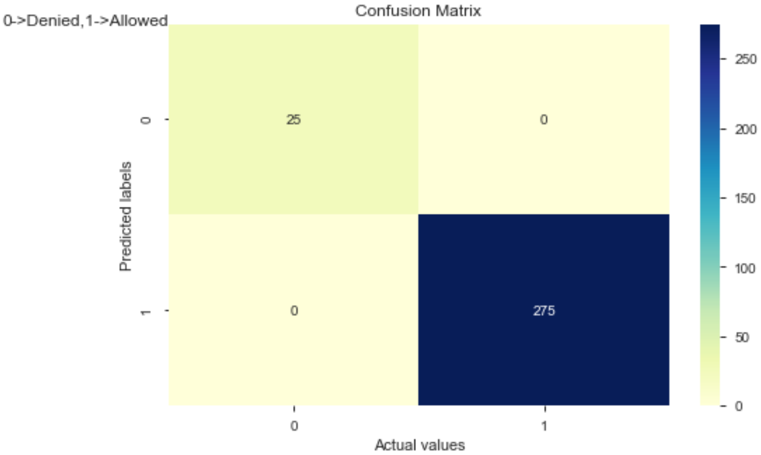


Figure 5.10 Confusion matrix-Actual and Predicted values for 300 Testdata

Confusion matrix shown in Figure 5.10 is generated to describe the performance of the proposed model. Classifiers made a total of 300 predictions, out of which 275 are allowed and 25 are denied.
Thus, the proposed model is able to perfectly classify the Labels.

5.6 PERFORMANCE ANALYSIS OF THE PREDICTION MODEL

The training phase of model terminates with the generation of a prediction model with benchmark dataset. For the purpose of testing the performance of the prediction model, the service broker acquires the workload trace data from the cloud service provider and fed into the prediction model. The performance of the prediction model is estimated through several performance metrics using Eqs (5.8) - (5.11),

$$Accuracy = \frac{(True\ Positive + True\ Negative)}{(True\ Positive + True\ Negative + False\ Positive + False\ Negative)} \tag{5.8}$$

$$Precision = \frac{(True\ Positive)}{(True\ Positive + False\ Positive)} \tag{5.9}$$

$$Recall = \frac{(True\ Positive)}{(True\ Positive + False\ Negative)} \tag{5.10}$$

$$F - Measure = 2 * \left(\frac{(Precision * Recall)}{(Precision + Recall)} \right) \tag{5.11}$$

The error values are estimated using Eqs (5.12) - (5.14)

$$Mean\ Absolute\ Error = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{5.12}$$

$$Mean\ Square\ Error = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{5.13}$$

$$Root\ Mean\ Square\ Error = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2}\ , \tag{5.14}$$

where $\widehat{y_i}$ and y_i are respective predicted and actual workloads at instance time i, while N is the number of samples. Lower the error value, higher the accuracy of the prediction

Table 5.3 Performance of the proposed model

Validation Parameters	Values
Accuracy	99.98%
Precision	99.98%
Recall	99.98%
F-Measure	99.80%
RMSE	0.182
MAE	0.033
MSE	0.033

model. The performance of the proposed model is given in Table 5.3. All the important prediction parameters such as Accuracy, Precision, Recall for the proposed prediction algorithm yield the value of 99.98% and F-measure is 99.8%.

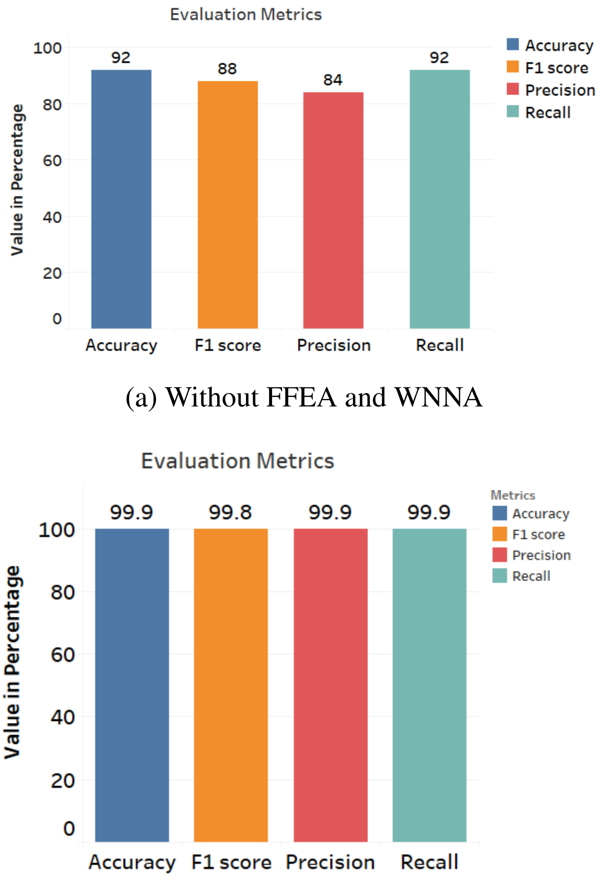
5.7 EVALUATION OF RESULTS

The proposed model is evaluated by comparing it with the existing models. Prediction error performance is presented here.

5.7.1 Performance validation of prediction parameters

Prediction values of the proposed model combining FFEA and WNNA is compared with the normal clustered test data, shown in the Figure 5.11.

We choose benchmark algorithms such as Logistics regression, Decision tree and Random forest. For the purpose of model comparison, we have chosen accuracy as a performance metric. Figure 5.12 shows the performance of the existing algorithms in the literature against the proposed model. From the graph it's evident that, for all the existing algorithms taken for comparision, only the Recall is reaching high values.



(a) Without FFEA and WNNA

(b) With FFEA and WNNA

Figure 5.11 Evaluation metrics obtained for Testdata

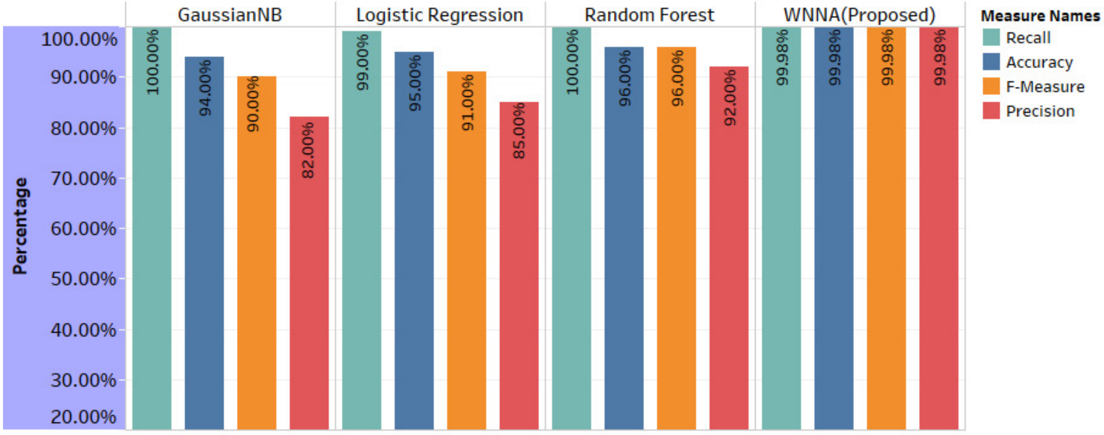
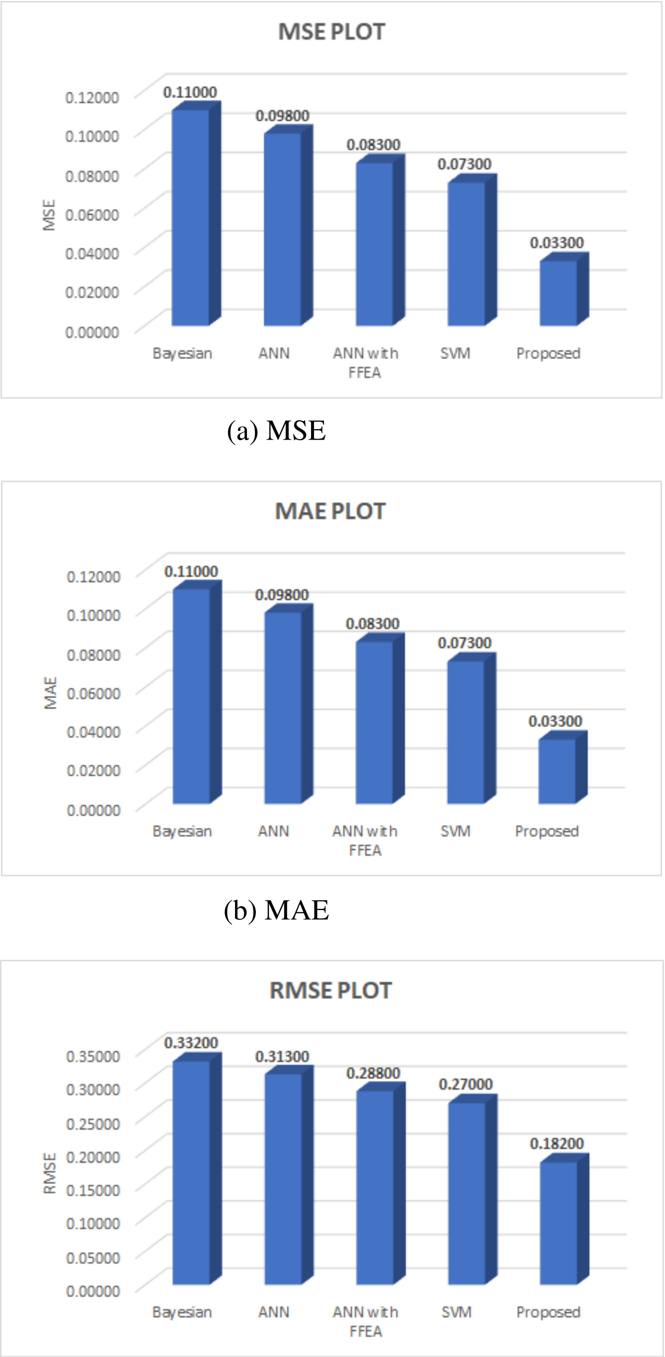


Figure 5.12 Evaluation metrics comparison

5.7.2 Error performance validation

WNNA prediction involves the estimation of three different forms of error and is compared with the existing models.



(a) MSE

(b) MAE

(c) RMSE

Figure 5.13 Validation of error parameters

The MSE and its root values are the most common error values in the prediction models. The mean error for the proposed model is estimated, and the comparison with the existing models are shown in the Figure 5.13.

Mean Square Error and Mean Absolute error values achieved by both the proposed and existing models are represented. From Figure 5.13, it is observed that the MSE and MAE value of the proposed model is 3.30% while Artificial Neural Network achieved 9.8% error. ANN with FFEA gave an error of 8.3% which is very close to the SVM model (Hu et al., 2013) having 7.3%. Bayesian model-based prediction (Shyam and Manvi, 2016) showed poor performance with MAE and MSE value of 11%.

Root Mean Square Error value for the proposed BeeM-NN model is valid with 18.2%, when compared to the Bayesian, ANN, ANN with FFEA and SVM with the error value 33.2%, 31.3%, 28.8% and 27% respectively. Overall, the proposed model performed well with less error values compared to other models.

5.8 SUMMARY

Proposed Workload Neural Network Algorithm (WNNA) framework and algorithm have been discussed in this chapter. Federated cloud workload predictor using CloudSim and Clustered data traces with and without FFEA are represented. Using statistical testing it is observed that the input features such as CPU, Memory and Disk impacts the output class label. Performance analysis of training and testing data is shown here. All the important prediction parameters such as Accuracy, precision, recall and F-measure are calculated. We have also calculated the error value between the predicted and actual data. The proposed model performed well with the minimal error and the better accuracy when compared to other benchmark models.

Chapter 6

NOVEL BEE MUTATION OPTIMIZATION METHOD

This chapter discusses the proposed Novel Bee Mutation Optimization Algorithm (NBMOA) model. Performance validation of the Service parameters are also discussed in this chapter.

6.1 PRELIMINARIES

6.1.1 Artificial Bee Colony

The ABC is an optimization algorithm that is nature-inspired by the foraging behavior of the honey bee. The ABC initially generates a randomized distributed workload solution population based on the size of a swarm (SS).

Components involved in ABC are:

- **Employed bees**- It looks for the food around the food supply,while exchanges the information about these sources with the onlooker bees.
- **Onlooker bees**-It appears to pick good sources of food from those collected by the previous bees.
- **Scout bees**- They are transformed from a few employed bees, leaving their sources of food and looking for new ones.
- The higher quality food source (fitness) would have a significant chance of being chosen by the onlooker bees.

Let $S_i = \{S_{i,1}, S_{i,2}, ..., S_{i,n}\}$ represent the i^{th} swarm solution with the dimension size D. Every generated bee S_i generates a new workload solution N_i in the neighbor position with S_k being selected randomly as candidate workload solution and j is a random

dimension index from the set $\{1, 2, ..., n\}$.
The corresponding workload solution is given in Eq (6.1) as,

$$N_{i,j} = S_{i,j} + \phi_{i,j} \left(s_{i,j} - s_{k,j} \right)_{(\mathrm{i} \neq k)(\phi_{i,j}) \varepsilon\ (-1,1)} \tag{6.1}$$

where, $\phi_{i,j}$ represents the Uniformly distributed random number which ranges between $[-1, 1]$. Greedy selection is performed to compare the values of S_i and N_i to update the workload solution based on better fitness value. When the computation of all bees is completed, the obtained information is transmitted to the onlooker bee. The selection of a workload solution is based on the probabilistic model given with the workload fitness function in Eq (6.2) as

$$P_i = \frac{F_i}{\sum_{j=1}^{SS} F_j}, \tag{6.2}$$

where F_i, F_j, are fitness measure of i^{th} , j^{th} swarm workload solution (Xu et al., 2013) respectively.

6.1.2 Genetic Algorithm

The Genetic Algorithm (GA) is a well-known optimizer that operates over coded chromosomes. The coding can be either decimal or binary. It is generally an iterative process which produces a theoretically better solution in each generation.

Entities involved in Genetic Algorithm are as follows:

- **Initial population**- set of individuals/solutions to the problem.
- **Fitness function**- It specifies how fit an individual is (the ability of an individual to compete with other individuals).
- **Selection**- In accordance with their fitness, pick two parent chromosomes from a population.
- **Crossover**-Cross parents with a new offspring.
- **Mutation**- It is a spontaneous chromosome tweak that also supports the concept of population diversity.

The GA usually involves three significant operators (Amjad et al., 2018). The selection operator chooses the items that satisfy the workload fitness function. The crossover operator provides two different offsprings that are distinctive from the two parent workload solutions. Finally, the mutation operator changes the solutions within its limit.

6.2 BEE MUTATION OPTIMIZATION

The predicted values of the workload are processed in the novel Bee mutation algorithm described in Algorithm 4, which is the hybridization of both Artificial Bee colony and Genetic algorithms.

The Bee optimization terminates with the estimation of the probability values for the entire bees. The selection operator of GA takes the probability values as input, and the rank-based selection is employed to get the optimum fitness. Initially, all the obtained probabilities of the solution are arranged in the order of low to high, and the ranking R is performed, then the rank value R_V in Eq (6.3) and performance value P_V in Eq (6.4) are estimated as,

$$R_V = 1 - \left(\frac{R}{P}\right) \tag{6.3}$$

$$P_V = \frac{Best\ Ranking\ in\ the\ solution\ population}{individual\ solution}, \tag{6.4}$$

where P is the total population of the bee. With both R_V and P_V values, the best value (G$_V$) is estimated as Eq (6.5)

$$G_V = R_V + P_V \tag{6.5}$$

From the goodness value G_V , the fitness value F_V is estimated in Eq (6.6) as,

$$F_V = \frac{G_V\ of\ individual\ solution}{\sum G_V\ \ in\ the\ solution\ population} \tag{6.6}$$

The selection operator selects the functions that possess higher fitness value and rejects low fitness values. The cross-over operator is executed over the two obtained fitness values to determine the cross-over probability (CO_P) with the following expression in Eq (6.7) as,

$$CO_P = \left\{ \begin{array}{l} if\ F_{Va} \leq F_{Vb},\ then\ K_1 \frac{F_{Vmax} - F_{Va}}{F_{Vmax} - F_{Vb}} \\ \\ if\ F_{Va} > F_{Vb},\ then\ K_2 \end{array} \right\}, \tag{6.7}$$

where F_{Va}, F_{Vb} are the maximum fitness value of two solutions and fitness average in solution population respectively, and F_{Vmax} is the complete solution to fitness value. K_1 and K_2 are the overall values which are in range of 0 to 1.

After the termination of the cross-over operator, the mutation operator is employed over its output. Similar to the cross-over, the mutation probability (M$_P$) is estimated in

Eq (6.8) as

$$M_P = \left\{ \begin{array}{l} if\ F_{Vc} \leq F_{Vb},\ then\ K_3 \frac{F_{Vmax} - F_{Vc}}{F_{Vmax} - F_{Vb}} \\ \\ if\ F_{Vc} > F_{Vb},\ then\ K_4 \end{array} \right\}, \tag{6.8}$$

where F_{Vc} is the individuals solution fitness and K_3 and K_4 which are in the range of 0 to 1.

Due to the operation of cross-over and mutation, a new breed of solution is generated, and it is re-evaluated for its fitness. The weak solution fitness values are replaced with better performing solution values. The process of the proposed optimization algorithm endures until it reaches the maximum cycle number.

6.3 EVALUATION OF RESULTS

The proposed model is evaluated by comparing it with the existing models.

6.3.1 Performance validation of optimization parameters

Cost and resource utilization are the important parameters considered during the optimization process.

The usage of RAM and CPU over various iteration of optimization over the predicted workload is plotted which is shown in Figure 6.1. From the mentioned figure, it is observed that the maximum CPU usage is 91.45% and the minimum usage is about 0.008% and the average usage is about 32.82%. Similarly, the RAM usage ranges from 28.87% to 40.82%.

Figure 6.1 also shows the cost incurred in each iteration. Cost decreases steadily with the increase in iteration.

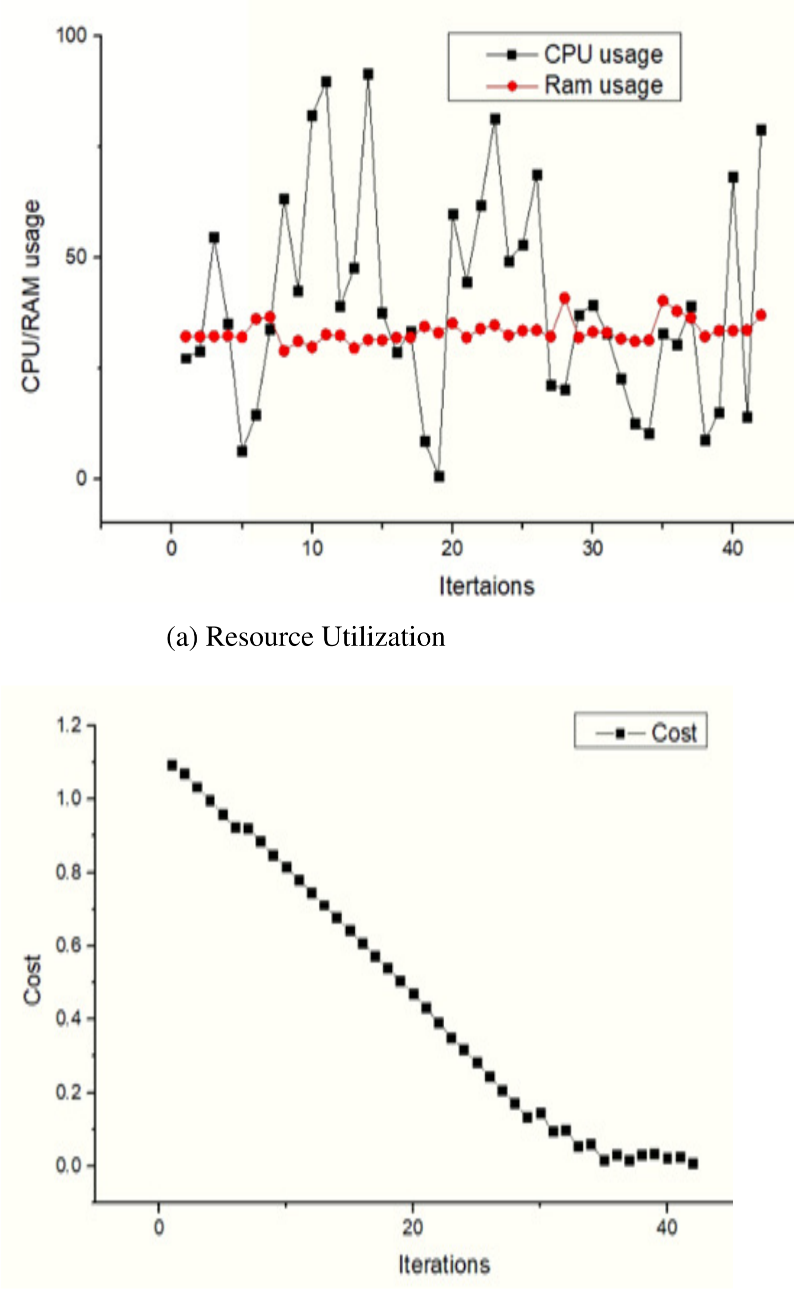
Cost for the proposed model is slightly lesser than the existing framework (DOCA) (Tavana et al., 2018) and GA, which is shown in Figure 6.2.

The resource utilization and cost is compared with the existing model as shown in Figure 6.2. The comparison shows that the proposed optimization is better than the existing framework (Levin et al., 2018).

Similarly, the time consumption of the proposed algorithm is nearly 25% better than the WWA (Arulkumar and Bhalaji, 2020) and 31.8% than ACO (Arulkumar and Bhalaji, 2020).

Algorithm 4 Novel Bee Mutate Optimizer Algorithm

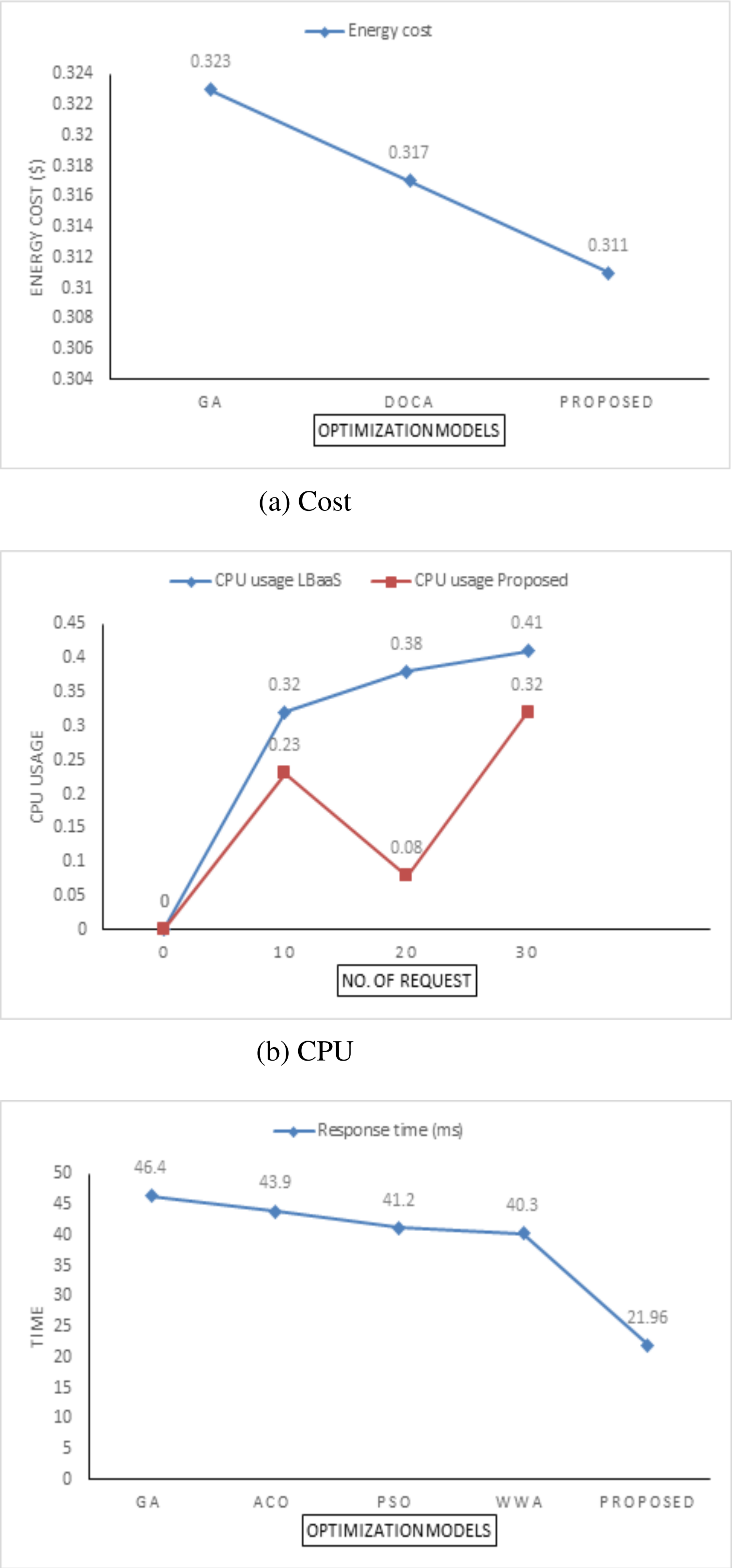
```
Procedure NBMOA(NS,n,MCN) NS-Number of Employed Bee, n-Search space, MCN-Maximum Cycle Number
for i ← 1 to NS do
    X(i,:) ← Random vector˙Initial population
    Fit(i) ← Fitness(X(i,:))˙Fitness function of X(i,:)
end for
[a,b] ← max(Fit)
Xbest ← X(b,:)
ct ← 1
while (True) do
    for i ← 1 to NS do
        for j ← 1 to n do
            $V(i,j) \leftarrow X(i,j) + \phi(i,j) * (X(i,j) - Xbest(j)).\phi(i,j) \in [-1,1]$
            Fit(i) ← Fitness(V (i,:))
        end for
    end for
    [a,b] ← max(Fit)
    if a > Xbest then
        Xbest ← a
    end if
end while
X← V
SumFit ← P(Fit)
SProb ← 0; P ← 0
for i← 1 to NS do
    Prob ← SProb + Fit(i) SumFit
    SProb ← P + Prob
end for
Crossover and mutation build a new generation and give birth to offspring
Assess the quality of individual offspring
Replacing the lowest ranked portion of the population with offspring
ct ← ct + 1
if ct>MCN then break
    end
end if
```



(a) Resource Utilization

(b) Cost

Figure 6.1 Resource utilization and cost over different iterations



(a) Cost

(b) CPU

(c) Time

Figure 6.2 Proposed model comparison on Cost, CPU and Response time

6.4 SUMMARY

A novel optimization technique is implemented in this chapter. Proposed algorithm uses Meta-heuristics techniques such as Genetic mutation and Artificial Bee Colony. Predicted values are fed into the Bee mutation algorithm in which the bee colony generates the probability of each generated solution. The performance of the proposed optimized prediction model is analyzed on optimization metrics. It is observed that the implemented model performs better than the other benchmark algorithms.

Chapter 7

CONCLUSION AND FUTURE WORK

This chapter summarizes the major contributions of the thesis. It also highlights the road-map for future research directions in the field of computing and optimization.

7.1 CONTRIBUTIONS

Thesis contributions are summarized as follows.

1. For efficient management of cloud workloads, a conceptual framework has been designed for a federated cloud environment.

2. Proposed a novel feature extraction technique to extract the important features from the raw workloads.

3. For predicting the future workloads, a novel prediction algorithm has been implemented using Artificial Neural Network.

4. Using Meta-heuristics technique, a novel optimization technique has been proposed to optimize the QoS parameters.

The first contribution is provided in **Chapter 3**, where all the modules are inter-dependent on each other. All the conceptual components are grouped into the framework and are provided in the mentioned chapter. Framework is mainly deigned with respect to cloud broker perspective. The important entities and their dependencies between each module are defined. Also, Architecture design and abstract design of each module are shown.

In the second contribution (**Chapter 4**), a novel fitness feature extraction Algorithm (FFEA) is implemented that extracts the important features from the raw dataset. The dataset is considered in the real datasets provided by the Microsoft Azure which is a

Publicly available traces. In the thesis, using the domain knowledge, we have considered CPU utilization, Memory storage and Disk space as the three important attributes. These features are considered to predict the future loads in the cloud environment. Statistical testing is carried to test the significance level of each attribute impacting on each other.

In the third contribution **(Chapter 5)**, Artificial Neural Network has been studied and using the model, a novel prediction algorithm (WNNA) has been implemented. Prediction framework is provided that consists of set up units and input and output layers. The developed model is tested with the data traces of workload from the cloud service provider. The error values in the training model were evaluated and reduced through the iterative process. The proposed model shows the better accuracy and the error values are very minimal compared to the other benchmark models.

In the fourth contribution **(Chapter 6)**, The novel architecture of BeeM-NN is introduced to the cloud computing environment for optimization based on the CPU utilization and storage spaces in disk & memory. Proposed Hybrid Optimization model consists of Ant Bee Colony and Genetic mutation. The obtained predicted values are fed into the Bee mutation algorithm in which the bee colony generates the probability of each generated solution. The obtained probability values underwent genetic operations to provide the optimized predicted value. The performance of the proposed optimized prediction model is analyzed on optimization metrics.

7.2 FUTURE RESEARCH DIRECTIONS

This section suggests some directions for possible future works. Several research directions are worth investigating as follows.

- The Cloud workload characterization is one of the important parts to be considered while computing in the cloud environment. The Time series continuous data sets can be considered with different multi attributes.

- The predicted workloads may be employed for resource allocation in different cloud deployment models. Scheduling policies can be implemented for the allotment for resources on different metrics.

- There are many optimization metrics involved during the computing process. Hence, the work can be extended by considering the network parameters such as latency and throughput.

REFERENCES

Amiri, M., Feizi-Derakhshi, M.-R., and Mohammad-Khanli, L. (2017). Ids fitted q improvement using fuzzy approach for resource provisioning in cloud. *Journal of Intelligent & Fuzzy Systems*, 32(1):229–240.

Amiri, M. and Mohammad-Khanli, L. (2017). Survey on prediction models of applications for resources provisioning in cloud. *Journal of Network and Computer Applications*, 82:93–113.

Amjad, M. K., Butt, S. I., Kousar, R., Ahmad, R., Agha, M. H., Faping, Z., Anjum, N., and Asgher, U. (2018). Recent research trends in genetic algorithm based flexible job shop scheduling problems. *Mathematical Problems in Engineering*, 2018.

Ardagna, D., Casolari, S., Colajanni, M., and Panicucci, B. (2012). Dual time-scale distributed capacity allocation and load redirect algorithms for cloud systems. *Journal of Parallel and Distributed Computing*, 72(6):796–808.

Arulkumar, V. and Bhalaji, N. (2020). Performance analysis of nature inspired load balancing algorithm in cloud environment. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–8.

Badger, L., Grance, T., Patt-Corner, R., Voas, J., et al. (2012). Cloud computing synopsis and recommendations. *NIST special publication*, 800:146.

Bagherzadeh, J. and MadadyarAdeh, M. (2009). An improved ant algorithm for grid scheduling problem. In *2009 14th International CSI Computer Conference*, pages 323–328. IEEE.

Bahga, A., Madisetti, V. K., et al. (2011). Synthetic workload generation for cloud computing applications. *Journal of Software Engineering and Applications*, 4(07):396.

Bohn, R. B., Messina, J., Liu, F., Tong, J., and Mao, J. (2011). Nist cloud computing reference architecture. In *2011 IEEE World Congress on Services*, pages 594–596. IEEE.

Boyd, W. T. and Recio, R. J. (1999). I/o workload characteristics of modern servers. In *Workload Characterization: Methodology and Case Studies, 1999*, pages 87–96. IEEE.

Breitgand, D., Marashini, A., and Tordsson, J. (2011). Policy-driven service placement optimization in federated clouds. *IBM Research Division, Tech. Rep*, 9:11–15.

Buyya, R., Ranjan, R., and Calheiros, R. N. (2010). Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services. In *International Conference on Algorithms and Architectures for Parallel Processing*, pages 13–31. Springer.

Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., and Brandic, I. (2009). Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, 25(6):599–616.

Cain, H. W., Rajwar, R., Marden, M., and Lipasti, M. H. (2001). An architectural evaluation of java tpc-w. In *High-Performance Computer Architecture, 2001. HPCA. The Seventh International Symposium on*, pages 229–240. IEEE.

Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., and Buyya, R. (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, 41(1):23–50.

Calheiros, R. N., Toosi, A. N., Vecchiola, C., and Buyya, R. (2012). A coordinator for scaling elastic applications across multiple clouds. *Future Generation Computer Systems*, 28(8):1350–1362.

Calzarossa, M. C., Della Vedova, M. L., Massari, L., Petcu, D., Tabash, M. I., and Tessera, D. (2016). Workloads in the clouds. In *Principles of Performance and Reliability Modeling and Evaluation*, pages 525–550. Springer.

Cetinski, K. and Juric, M. B. (2015). Ame-wpc: Advanced model for efficient workload prediction in the cloud. *Journal of Network and Computer Applications*, 55:191–201.

Chaisiri, S., Lee, B.-S., and Niyato, D. (2011). Optimization of resource provisioning cost in cloud computing. *IEEE transactions on services Computing*, 5(2):164–177.

Chang, Y.-C., Chang, R.-S., and Chuang, F.-W. (2014). A predictive method for workload forecasting in the cloud environment. In *Advanced Technologies, Embedded and Multimedia for Human-Centric Computing*, pages 577–585. Springer.

Chen, C., He, B., and Tang, X. (2012). Green-aware workload scheduling in geographically distributed data centers. In *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, pages 82–89. IEEE.

Chen, W.-n., Shi, Y., and Zhang, J. (2009). An ant colony optimization algorithm for the time-varying workflow scheduling problem in grids. In *2009 IEEE Congress on Evolutionary Computation*, pages 875–880. IEEE.

Chiang, C.-W., Lee, Y.-C., Lee, C.-N., and Chou, T.-Y. (2006). Ant colony optimisation for task matching and scheduling. *IEE Proceedings-Computers and Digital Techniques*, 153(6):373–380.

Clark, M., Durg, A., and Lienenbrugger, K. (2001). Characterization of tpc-h queries on amd athlon/sup tm/microprocessors. In *Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop on*, pages 26–35. IEEE.

de Oliveira, G. S., Ribeiro, E., Ferreira, D. A., Araújo, A. P., Holanda, M. T., and Walter, M. E. M. (2013). Acosched: a scheduling algorithm in a federated cloud infrastructure for bioinformatics applications. In *Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on*, pages 8–14. IEEE.

Dorigo, M. and Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical computer science*, 344(2-3):243–278.

Gao, Y., Wang, Y., Gupta, S. K., and Pedram, M. (2013). An energy and deadline aware resource provisioning, scheduling and optimization framework for cloud systems. In *Proceedings of the Ninth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, page 31. IEEE Press.

Garg, S. K., Gopalaiyengar, S. K., and Buyya, R. (2011). Sla-based resource provisioning for heterogeneous workloads in a virtualized cloud datacenter. In *international conference on Algorithms and architectures for parallel processing*, pages 371–384. Springer.

Guérout, T., Monteil, T., Da Costa, G., Calheiros, R. N., Buyya, R., and Alexandru, M. (2013). Energy-aware simulation with dvfs. *Simulation Modelling Practice and Theory*, 39:76–91.

Gupta, S. and Dinesh, D. A. (2017). Resource usage prediction of cloud workloads using deep bidirectional long short term memory networks. In *2017 IEEE International*

Conference on Advanced Networks and Telecommunications Systems (ANTS), pages 1–6. IEEE.

Gupta, S., Muthiyan, N., Kumar, S., Nigam, A., and Dinesh, D. A. (2017). A supervised deep learning framework for proactive anomaly detection in cloud workloads. In *2017 14th IEEE India Council International Conference (INDICON)*, pages 1–6. IEEE.

Hongyuan, Z., Liang, Z., and Jiaqi, W. (2006). Design and implementation of load balancing in web server cluster system [j]. *Journal of Nanjing University of Aeronautics & Astronautics*, 3.

Howell, F. and McNab, R. (1998). Simjava: A discrete event simulation library for java. *Simulation Series*, 30:51–56.

Hu, R., Jiang, J., Liu, G., and Wang, L. (2013). Cpu load prediction using support vector regression and kalman smoother for cloud. In *2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops*, pages 88–92. IEEE.

Huang, Z., Peng, J., Lian, H., Guo, J., and Qiu, W. (2017). Deep recurrent model for server load and performance prediction in data center. *Complexity*, 2017.

Huu, T. N., Ngoc, N. P., Thu, H. T., Ngoc, T. T., Minh, D. N., Tai, H. N., Quynh, T. N., Hock, D., Schwartz, C., et al. (2013). Modeling and experimenting combined smart sleep and power scaling algorithms in energy-aware data center networks. *Simulation Modelling Practice and Theory*, 39:20–40.

Islam, S., Keung, J., Lee, K., and Liu, A. (2012). Empirical prediction models for adaptive resource provisioning in the cloud. *Future Generation Computer Systems*, 28(1):155–162.

Javadi, B., Abawajy, J., and Buyya, R. (2012). Failure-aware resource provisioning for hybrid cloud infrastructure. *Journal of parallel and distributed computing*, 72(10):1318–1331.

John, L. K., Vasudevan, P., and Sabarinathan, J. (1999). Workload characterization: Motivation, goals and methodology. In *Workload Characterization: Methodology and Case Studies, 1999*, pages 3–14. IEEE.

Kang, H., Chen, Y., Wong, J. L., Sion, R., and Wu, J. (2011). Enhancement of xen's scheduler for mapreduce workloads. In *Proceedings of the 20th international symposium on High performance distributed computing*, pages 251–262. ACM.

Karimi, M. B., Isazadeh, A., and Rahmani, A. M. (2017). Qos-aware service composition in cloud computing using data mining techniques and genetic algorithm. *The Journal of Supercomputing*, 73(4):1387–1415.

Kashan, A. H. (2009). League championship algorithm: a new algorithm for numerical function optimization. In *2009 international conference of soft computing and pattern recognition*, pages 43–48. IEEE.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95-International Conference on Neural Networks*, volume 4, pages 1942–1948. IEEE.

Khan, A., Yan, X., Tao, S., and Anerousis, N. (2012). Workload characterization and prediction in the cloud: A multiple time series approach. In *2012 IEEE Network Operations and Management Symposium*, pages 1287–1294. IEEE.

Kim, J.-S., Qin, X., and Hsu, Y. (1999). Memory characterization of a parallel data mining workload. In *Workload Characterization: Methodology and Case Studies, 1999*, pages 60–68. IEEE.

Kochut, A. and Beaty, K. (2007). On strategies for dynamic resource management in virtualized server environments. In *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2007. MASCOTS'07. 15th International Symposium on*, pages 193–200. IEEE.

Kousalya, K. and Balasubramanie, P. (2009). To improve ant algorithm's grid scheduling using local search. *Int. J. Comput. Cogn*, 7(4):47–57.

Levin, A., Lorenz, D., Merlino, G., Panarello, A., Puliafito, A., and Tricomi, G. (2018). Hierarchical load balancing as a service for federated cloud networks. *Computer communications*, 129:125–137.

Li, S., Ren, S., Yu, Y., Wang, X., Wang, L., and Quan, G. (2012). Profit and penalty aware scheduling for real-time online services. *IEEE Transactions on industrial informatics*, 8(1):78–89.

Lin, J.-C., Wu, C.-H., and Wei, W.-L. (2011). Error weighted semi-coupled hidden markov model for audio-visual emotion recognition. *IEEE Transactions on Multimedia*, 14(1):142–156.

Liu, A. and Wang, Z. (2008). Grid task scheduling based on adaptive ant colony algorithm. In *2008 International conference on management of e-commerce and e-government*, pages 415–418. IEEE.

Liu, H., Abraham, A., and Hassanien, A. E. (2010). Scheduling jobs on computational grids using a fuzzy particle swarm optimization algorithm. *Future Generation Computer Systems*, 26(8):1336–1343.

Lucas-Simarro, J. L., Moreno-Vozmediano, R., Montero, R. S., and Llorente, I. M. (2013). Scheduling strategies for optimal service deployment across multiple clouds. *Future Generation Computer Systems*, 29(6):1431–1441.

Marcus, R. and Papaemmanouil, O. (2016a). Wisedb: A learning-based workload management advisor for cloud databases. *Proceedings of the VLDB Endowment*, 9(10):780–791.

Marcus, R. and Papaemmanouil, O. (2016b). Wisedb: A learning-based workload management advisor for cloud databases. *Proc. VLDB Endow.*, 9(10):780–791.

Mathiyalagan, P., Suriya, S., and Sivanandam, S. (2011). Hybrid enhanced ant colony algorithm and enhanced bee colony algorithm for grid scheduling. *International Journal of Grid and Utility Computing*, 2(1):45–58.

Microsoft, D. (2019). Microsoft azure public dataset cpu readings trace data /azurepublicdataset.

Morariu, O., Morariu, C., and Borangiu, T. (2012). A genetic algorithm for workload scheduling in cloud based e-learning. In *Proceedings of the 2nd International Workshop on Cloud Computing Platforms*, page 5. ACM.

Moreno-Vozmediano, R., Montero, R. S., and Llorente, I. M. (2012). Iaas cloud architecture: From virtualized datacenters to federated cloud infrastructures. *Computer*, 45(12):65–72.

Mulia, W. D., Sehgal, N., Sohoni, S., Acken, J. M., Stanberry, C. L., and Fritz, D. J. (2013). Cloud workload characterization. *IETE Technical Review*, 30(5):382–397.

Murta, C. D. and Almeida, V. A. (1999). Characterizing response time of www caching proxy servers. In *Workload Characterization: Methodology and Case Studies, 1999*, pages 69–75. IEEE.

Nan, X., He, Y., and Guan, L. (2013). Optimization of workload scheduling for multimedia cloud computing. In *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, pages 2872–2875. IEEE.

Nogueira, D., Rocha, L., Santos, J., Araújo, P., Almeida, V., and Meira, W. (2002). A methodology for workload characterization of filesharing peer-to-peer networks. In *WWC'02: Proceedings of the 5th IEEE International Workshop on Workload Characterization*.

Pacini, E., Mateos, C., and García Garino, C. (2014). Dynamic scheduling based on particle swarm optimization for cloud-based scientific experiments. *CLEI Electronic Journal*, 17(1):3–3.

Paton, N., De Aragão, M. A., Lee, K., Fernandes, A. A., and Sakellariou, R. (2009). Optimizing utility in cloud computing through autonomic workload execution. *Bulletin of the Technical Committee on Data Engineering*, 32(1):51–58.

Pizzuti, C. (2011). A multiobjective genetic algorithm to find communities in complex networks. *IEEE Transactions on Evolutionary Computation*, 16(3):418–430.

Qiu, F., Zhang, B., and Guo, J. (2016). A deep learning approach for vm workload prediction in the cloud. In *2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pages 319–324. IEEE.

Qiu, X., Yeow, W. L., Wu, C., and Lau, F. C. (2013). Cost-minimizing preemptive scheduling of mapreduce workloads on hybrid clouds. In *Quality of Service (IWQoS), 2013 IEEE/ACM 21st International Symposium on*, pages 1–6. IEEE.

Roy, N., Dubey, A., and Gokhale, A. (2011). Efficient autoscaling in the cloud using predictive models for workload forecasting. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 500–507. IEEE.

Sahi, S. K. and Dhaka, V. (2016). A survey paper on workload prediction requirements of cloud computing. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 254–258. IEEE.

Sakellari, G. and Loukas, G. (2013). A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing. *Simulation Modelling Practice and Theory*, 39:92–103.

Salehinejad, H., Sankar, S., Barfett, J., Colak, E., and Valaee, S. (2017). Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*.

Sarikaya, R., Isci, C., and Buyuktosunoglu, A. (2010). Runtime workload behavior prediction using statistical metric modeling with application to dynamic power management. In *IEEE International Symposium on Workload Characterization (IISWC'10)*, pages 1–10. IEEE.

Sathappan, O., Chitra, P., Venkatesh, P., and Prabhu, M. (2011). Modified genetic algorithm for multiobjective task scheduling on heterogeneous computing system. *International Journal of Information Technology, Communications and Convergence*, 1(2):146–158.

Seshadri, P. and Mericas, A. (2001). Workload characterization of multithreaded java servers on two powerpc processors. In *Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop on*, pages 36–44. IEEE.

Shishira, S. and Kandasamy, A. (2020a). Beem-nn: An efficient workload optimization using bee mutation neural network in federated cloud environment. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–17.

Shishira, S. and Kandasamy, A. (2020b). A comprehensive survey on federated cloud computing and its future research directions. *Evolutionary Computing and Mobile Sustainable Networks*, pages 79–88.

Shyam, G. K. and Manvi, S. S. (2016). Virtual resource prediction in cloud environment: a bayesian approach. *Journal of Network and Computer Applications*, 65:144–154.

Song, B., Yu, Y., Zhou, Y., Wang, Z., and Du, S. (2018). Host load prediction with long short-term memory in cloud computing. *The Journal of Supercomputing*, 74(12):6554–6568.

Sun, J., Xiong, S.-W., and Guo, F.-M. (2004). A new pheromone updating strategy in ant colony optimization. In *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 04EX826)*, volume 1, pages 620–625. IEEE.

Sun, Y. S., Chen, Y.-F., and Chen, M. C. (2013). A workload analysis of live event broadcast service in cloud. *Procedia Computer Science*, 19:1028–1033.

Tavana, M., Shahdi-Pashaki, S., Teymourian, E., Santos-Arteaga, F. J., and Komaki, M. (2018). A discrete cuckoo optimization algorithm for consolidation in cloud computing. *Computers & Industrial Engineering*, 115:495–511.

Tian, W., Xue, R., Cao, J., Xiong, Q., and Hu, Y. (2013). An energy-efficient online parallel scheduling algorithm for cloud data centers. In *2013 IEEE Ninth World Congress on Services*, pages 397–402. IEEE.

Tian, W., Zhao, Y., Zhong, Y., Xu, M., and Jing, C. (2011). A dynamic and integrated load-balancing scheduling algorithm for cloud datacenters. In *2011 IEEE International Conference on Cloud Computing and Intelligence Systems*, pages 311–315. IEEE.

Tordsson, J., Montero, R. S., Moreno-Vozmediano, R., and Llorente, I. M. (2012). Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. *Future generation computer systems*, 28(2):358–367.

Urgaonkar, R., Wang, S., He, T., Zafer, M., Chan, K., and Leung, K. K. (2015). Dynamic service migration and workload scheduling in edge-clouds. *Performance Evaluation*, 91:205–228.

Van den Bossche, R., Vanmechelen, K., and Broeckhove, J. (2010). Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads. In *2010 IEEE 3rd international conference on cloud computing*, pages 228–235. IEEE.

Van den Bossche, R., Vanmechelen, K., and Broeckhove, J. (2013). Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds. *Future Generation Computer Systems*, 29(4):973–985.

Vecchiola, C., Calheiros, R. N., Karunamoorthy, D., and Buyya, R. (2012). Deadline-driven provisioning of resources for scientific applications in hybrid clouds with aneka. *Future Generation Computer Systems*, 28(1):58–65.

Vercauteren, T., Aggarwal, P., Wang, X., and Li, T.-H. (2007). Hierarchical forecasting of web server workload using sequential monte carlo training. *IEEE transactions on signal processing*, 55(4):1286–1297.

Wen, X., Huang, M., and Shi, J. (2012). Study on resources scheduling based on aco allgorithm and pso algorithm in cloud computing. In *2012 11th International Symposium on Distributed Computing and Applications to Business, Engineering & Science*, pages 219–222. IEEE.

Wright, P., Sun, Y. L., Harmer, T., Keenan, A., Stewart, A., and Perrott, R. (2012). A constraints-based resource discovery model for multi-provider cloud environments. *Journal of cloud computing: advances, systems and applications*, 1(1):6.

Xiao, Z., Song, W., and Chen, Q. (2012). Dynamic resource allocation using virtual machines for cloud computing environment. *IEEE transactions on parallel and distributed systems*, 24(6):1107–1117.

Xing, L.-N., Chen, Y.-W., Wang, P., Zhao, Q.-S., and Xiong, J. (2010). A knowledge-based ant colony optimization for flexible job shop scheduling problems. *Applied Soft Computing*, 10(3):888–896.

Xu, Y., Fan, P., and Yuan, L. (2013). A simple and efficient artificial bee colony algorithm. *Mathematical Problems in Engineering*, 2013.

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

Yang, L., Foster, I., and Schopf, J. M. (2003). Homeostatic and tendency-based cpu load predictions. In *Proceedings International Parallel and Distributed Processing Symposium*, pages 9–pp. IEEE.

Youseff, L., Butrico, M., and Da Silva, D. (2008). Toward a unified ontology of cloud computing. In *2008 Grid Computing Environments Workshop*, pages 1–10. IEEE.

Yuan, H., Bi, J., Tan, W., and Li, B. H. (2016). Cawsac: Cost-aware workload scheduling and admission control for distributed cloud data centers. *IEEE Transactions on Automation Science and Engineering*, 13(2):976–985.

Zeng, N., Wang, Z., Zhang, H., and Alsaadi, F. E. (2016). A novel switching delayed pso algorithm for estimating unknown parameters of lateral flow immunoassay. *Cognitive Computation*, 8(2):143–152.

Zeng, N., Wang, Z., Zhang, H., Kim, K.-E., Li, Y., and Liu, X. (2019). An improved particle filter with a novel hybrid proposal distribution for quantitative analysis of gold immunochromatographic strips. *IEEE Transactions on Nanotechnology*, 18:819–829.

Zeng, N., Wang, Z., Zineddin, B., Li, Y., Du, M., Xiao, L., Liu, X., and Young, T. (2014). Image-based quantitative analysis of gold immunochromatographic strip via cellular neural network approach. *IEEE transactions on medical imaging*, 33(5):1129–1136.

Zhang, F., Cao, J., Tan, W., Khan, S. U., Li, K., and Zomaya, A. Y. (2014). Evolutionary scheduling of dynamic multitasking workloads for big-data analytics in elastic cloud. *IEEE Transactions on Emerging Topics in Computing*, 2(3):338–351.

Zhang, H., Jiang, G., Yoshihira, K., Chen, H., and Saxena, A. (2009). Intelligent workload factoring for a hybrid cloud computing model. In *Services-I, 2009 World Conference on*, pages 701–708. IEEE.

Zhang, Y., Fan, W.-P., Wu, X., Chen, H., Li, B.-Y., and Zhang, M.-L. (2019). Cafe: Adaptive vdi workload prediction with multi-grained features. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5821–5828.

Zhou, Y., Xiang, Y., Chen, Z., He, J., and Wang, J. (2018). A scalar projection and angle-based evolutionary algorithm for many-objective optimization problems. *IEEE transactions on cybernetics*, 49(6):2073–2084.

Zhu, Y., Zhang, W., Chen, Y., and Gao, H. (2019). A novel approach to workload prediction using attention-based lstm encoder-decoder network in cloud environment. *EURASIP Journal on Wireless Communications and Networking*, 2019(1):274.

Zou, W., Zhu, Y., Chen, H., and Zhang, B. (2011). Solving multiobjective optimization problems using artificial bee colony algorithm. *Discrete dynamics in nature and society*, 2011.

LIST OF PUBLICATIONS /COMMUNICATIONS BASED ON THESIS:

Journal papers

- Shishira S R, and A. Kandasamy: BeeM-NN: An efficient workload optimization using Bee Mutation Neural Network in federated cloud environment. **Journal of Ambient Intelligence and Humanized Computing**, Springer (2020) pp.1-17. (SCIE)
- Shishira S R, A. Kandasamy: Ontology based Context-Aware Model for Intelligent Scheduling in Federated Cloud. **International Journal of Future Generation Communication and Networking**, vol.13, 2020. (Web of Science)
- Shishira S R, A. Kandasamy: A Novel Feature Extraction Model for Largescale Workload Prediction in Cloud Environment. **SN Computer Science**, vol.5, 2021, Springer. (Scopus)
- Shishira S R, A. Kandasamy: An Enhanced Neural Network model with Novel Feature Extraction for Efficient Load Forecasting in Cloud. **Journal of Ambient Intelligence and Humanized Computing**, Springer. (Under Review)

Conference papers

- Shishira SR, A. Kandasamy, K. Chandrasekaran: Survey on Meta Heuristics Optimization Techniques in Cloud Computing. 5th International Conference on Advances in Computing, Communications and Informatics, LNMIIT Jaipur, India, September 21-24, 2016, IEEE. (Scopus)
https://doi.org/10.1109/ICACCI.2016.7732249
- Shishira SR, A. Kandasamy, K. Chandrasekaran: Workload Scheduling in Cloud: A Comprehensive Survey and Future Research Directions. 7th International Conference on Cloud Computing, Data Science & Engineering, CONFLUENCE'17, Amity University, Noida, India, January 12-13, 2017, IEEE. (Scopus)
https://doi.org/10.1109/CONFLUENCE.2017.7943161
- Shishira SR, A. Kandasamy, K. Chandrasekaran: Workload Characterization: Survey of Current Approaches and Research Challenges. 7th International Conference on Computer and Communication Technology, November 24-26, 2017,

MNNIT Allahabad, India, ACM. (Scopus)
https://doi.org/10.1145/3154979.3155003

- Shishira SR, A. Kandasamy, K. Chandrasekaran: Comparative Study of Simulation Tools and Challenging Issues in Cloud Computing. International Conference on Intelligent Information Technologies, CEG Anna University, December 20 - 22, 2017, SPRINGER. (Scopus)
https://doi.org/10.1007/978-981-10-7635-0_1

- Shishira SR, A. Kandasamy: A Comprehensive Survey on Federated Cloud Computing and its Future Research Directions. International Conference on Evolutionary Computing and Mobile Sustainable Networks, Sir.MVIT, Bangalore, Feb 20-21, 2020, SPRINGER. (Scopus)
https://doi.org/10.1007/978-981-15-5258-8_9

- Shishira SR, A. Kandasamy: Conceptual Framework for Intelligent Management of Workloads in Cloud Environment. International Conference on Computing Methodologies and Communication, March 11-13, 2020, IEEE. (Scopus)
https://doi.org/10.1109/ICCMC48092.2020.ICCMC-0006

BIO-DATA

Name	:	Shishira S R
Email Id	:	shishirasr@gmail.com
Mobile	:	+91-8904029249
Date of Birth	:	April 10, 1990
Address	:	D/o. H. G. Ramachandra, # 4-80/7, "SANMATHI", Mahathma Nagar Layout, Kavoor post Bondel, Managlore-575015. Karnataka, India.

Educational Qualifications:

Degree	Year of Passing	University
BE (CSE)	2012	Visvesvaraya Technological University, Belgaum.
M-Tech (CN)	2014	Visvesvaraya Technological University, Belgaum.