Investigating Relationship between temperature, power and CPU load in cloud server

Aviral Bhandari

Thesis submitted for the degree of
Master in Network and System Administration

30 credits

Department of Informatics

Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2018

Investigating Relationship between temperature, power and CPU load in cloud server

Aviral Bhandari

Abstract

High power consumption has been a major issue for the data center. Higher consumption of power also results in higher concentration of greenhouse gas emission. A lot of research has been conducted to keep the operating temperature of the data center within certain limits. At the end of the day, a penny saved is a penny earned. Cooling costs account for in average about 40 percent of power bill of the data centers. The cooling cost of the IT infrastructure can be higher than operating them. The general theory is operating the server on lower temperature will reduce the cooling costs which in turn will reduce the electricity bill.

In this thesis, we will be researching if there is any relationship between temperatures, power consumption, and CPU utilization within data centers. We will be attempting to discover the relationship between the internal temperature of the server and its power consumption. We will be utilizing the real data collected from production server ALTO over the period of 11 days.

To accomplish our objective we will be using correlation as a statistical tool. We will also be developing supervised machine learning model and fit our data in the model with some most popular machine learning algorithms. The degree on how well the data fits into the machine learning model along with results from correlation tests will be used as means to access if there is a mappable relation between internal temperatures within the server and power consumption.

Acknowledgements

First of all, I would like to express my warmest gratitude to the University of Oslo and Oslo metropolitan University (formerly Oslo and Akershus University college) for providing me with such a glorious opportunity to be part of network and system administration course.

I am very much thankful to my mother, Janaki Baral for always being supportive of me against all the odds and hardships. She is, in fact, my real superhero.

This thesis could not have been completed without proper supervision of my supervisor Andre Brodtkorb. His guidance, ideas, and feedback were key to successful completion of the thesis. He provided me with the valuable suggestion on how to approach the project and often troubleshoot issues I face with my scripts during the thesis. I don't know how can I thank him for all his valuable effort. Simple words of Thanks will be just too less for what he had done for me.

I want to express my sincere thanks to my co-supervisor Hårek Haugerud. He was the man behind the scene running my scripts in ALTO and providing me with data and results. He also provisioned me with the test server Research5. Without him, there would have been no project. I am indebted to all his efforts behind the scene for completion of this project.

I would also like to express my appreciation to my friend Khagendra basnet for sharing his work in IBM bluemix and helping me to set up and troubleshoot issues related to Bluemix. IBM cloud platform used in this thesis is based on his work in his thesis last year.I was quite unlucky with my working laptop. It broke 4 times during the thesis. I am thankful to my friend Ramesh Upreti for providing me his laptop to work when my laptop broke at a critical stage of the project and was gone for repair.This project would not have been completed it was not for him to provide me with a laptop.He also helped me in debugging some of the silly errors that I made in the script.

I would like to thank my dear sister Madhavi Bhandari for correcting grammatical error in one of the chapter.

I have gained knowledge by going through a lot of open forums like StackOverflow, Quora and. Videos on youtube from various content creator like sentex, khan academy, udemy etc. provided me with the better understanding of what I was doing. I would like to shout a word of appreciation for all their hard work and willingness to openly share their knowledge with others.

There are various people who directly or indirectly involved during this project and I am thankful for all of them.

Sincerly
Aviral

Contents

List of Figures

List of Tables

Acronyms

The following acronyms are used in this thesis work:
CF → *Cloud Foundry*
IoT → *Internet of Things*
ACL → *Access control list*
TIA → *Telecommunication Industry Association*
CPU → *Central Processing Unit*
IaaS → *Infrastructure as service*
PaaS → *Platform as service*
SaaS → *Software as service*
MQTT → *Message Queuing Telemetry Transport*
TLS → *Transport layer security*
VM → *Virtual Machine*
NUMA → *Non Uniform Memory Access*
SMP → *Symmetric Multiprocessing*
AI → *Artificial Intelligence*
ML → *Machine learning*
SVM → *Support vector machine*
RF → *Rain Forest*
DT → *Decision Tree*
ANN → *Artificial Neural Network*
OSPF → *Open shortest path first*
EIGRP → *Enhanced Interior Gateway Routing Protocol*
CSV → *Comma seperated values*
GPU → *Graphics Processing Unit*

Part I

Introduction

Chapter 1

Introduction

Cloud computing has evolved to be one of the hottest technology in last decade or so. With the large companies opting for cloud based solution there has been tremendous increase in the amount of data that cloud processes. With a predicted explosion of IoT within few years,the data volume that cloud processes will significantly rise. Cisco predicts 50 billion connected devices by 2020. That number is expected to increase by five fold within 2025. With all of these devices producing data of their own, the aggregated volume of the data will be much more than ever seen. Connected cars which is going to enable the next generation autonomous car will be sending 25 GB of data to the cloud every hours.

We are using data centers in our daily life directly or indirectly. From posting a status on Facebook to swiping the debit card in the shopping mall to purchase goods and getting direction on the GPS navigator, each of this activities uses a data center. The data centers are now an integral part of our daily life. Data centers need to run day and night, 24 hours a day, 365 days a year to process and store information. As a result, data centers are one of largest consumers of electricity. In 2014 data centers consumed 70 billion kilowatt-hours of electricity in the US alone [41]. The power consumption of data centers is greater than a country like Argentina [11]. Heat generated by data centers is a major issue. Cooling consumes almost 40 percent or more of the data center's total energy consumption. It has been noted that sometimes the cooling cost of the IT infrastructures in the data center can be higher than operating cost of those infrastructures.

There is no wonder that reduction of heat generation or cutting off the cooling power consumption can significantly improve the overall costs of running the data centers. Many works have focused on way to reduce the power consumption of the data centers, few of them are discussed in **??**. Some works have been focused on thermal-aware workload distribution to reduce the operating temperatures.

In this thesis, we attempt to find a relationship if it exists between power consumption and temperature in cloud-based server and if thermal-aware workload scheduling really saves the power bill. We will be analyzing the real life data from the production server to achieve it. We will also be extending the Khagendra's work [3] to build very simple monitoring system over IBM cloud

1.1 Problem statement

The question for which we are striving to find an answer in this thesis is:
"How can we find relationship between internal temperature,power consumption and CPU load in a cloud based server?"

Internal temperature and power consumptions are the temperature and power consumptions reported by the system. Internal temperatures are the temperatures of different areas within the motherboard and power is the power required to operate the server. The power consumption we are taking into consideration does not account for the power required by the networking infrastructure such as switch and router. Power consumption mentioned throughout this thesis is the power consumption of the computing , not the networking.

The server is the device or program that provides functionality to other programs or users. The functionality may refer anything from accessing the data on file stored to performing a complex computation on data.

The cloud-based server is logically built on the top of the physical server by the means of virtualization. The cloud-based servers are hosted and delivered through cloud computing platforms over the internet.

Virtualization is the creation of virtual computers rather than an actual one such as OS, storage devices or network resources. It allows a single physical instance of the resources such as disk drive to be shared among multiple user or programs. Hypervisor takes care of mapping logical resources to a physical one. A hypervisor is a function which abstracts – isolates – operating systems and applications from the underlying computer hardware[5].

Temperature mentioned throughout the thesis is the internal temperature unless otherwise stated. There are 64 cores(32 logical and 32 physical) in the both the servers used in this thesis. We will refer to these cores as CPU followed by their ID assigned by the Operating system from CPU0 to CPU63. By stating CPU(X) where X is the cpuid from 0-63, we are referring to CPU load of CPU with id X unless stated otherwise. For eg. the notation of CPU0 should be interpreted as CPU utilization of CPU with if 0 i.e core0

Chapter 2

Background

2.1 Data center

The data center is the department in an enterprise that houses and maintains back-end information technology (IT) systems and data stores—its mainframes, servers, and databases. In the days of large, centralized IT operations, this department and all the systems resided in one physical place, hence the name data center[19].

The data center is any room or place where that houses computing facilities like servers storage units, routers, switches, and firewalls, as well as supporting components like backup power supply, fire suppression facilities and air conditioning. Data centers need to run around the clock 24 hours a day. Data centers have multiple clients which share the resources of the data centers. Some Large organization like Google, Apple have the data center of their own. The primary function of the data center is to provide its customer and users with various computing resources and storage or data backup services also perform the complex task like data analytics. There are four major components of the data center[7].

1. *White space*
   White space is the area of usable raised floor environment in square feet. For data centers that don't use raised environment, the term refers to a usable area.
2. *Support infrastructure*
   Support infrastructure refers to equipment required for smooth operation of the data centers. This includes power transformers, UPS, computer room air conditioners (CRAC) etc. In class 3 data center support infrastructure consume 4 times more space than white space.
3. *IT Infrastructure*

IT Infrastructure covers all the hardware with Racks hosting them along with cables and networking equipment

4. *Operations*
   Operations staffs are the group of people who are responsible for running and maintaining data centers. Primary tasks involve operating, maintaining, upgrading and repairing data centers when necessary

2.1.1 Data Center Architecture

A prevalence of the data centers follows layered architecture first purposed by Cisco. Layered architecture is tested and improved over past several years across multiple data centers. Layered architecture has some notable gains such as scalability, performance, flexibility, load distribution, redundancy, resiliency, and maintenance. Cisco defines 3 different layers as:

- *Core layer*
  It is the backbone of the network. It offers high speed, low latency packet switching to get the traffic in and out of data centers and to other networks. The core layer switches and routers are usually the most powerful. Core layer routers run interior routing protocols like OSPF and EIGRP [1] and provide load balancing between core and distribution layer using some hashing algorithm. One of the key requirement of the core layer is a fast convergence[2] so the triangle topology is highly favored to the square topology. Fig 2.2 shows the comparison of these design.

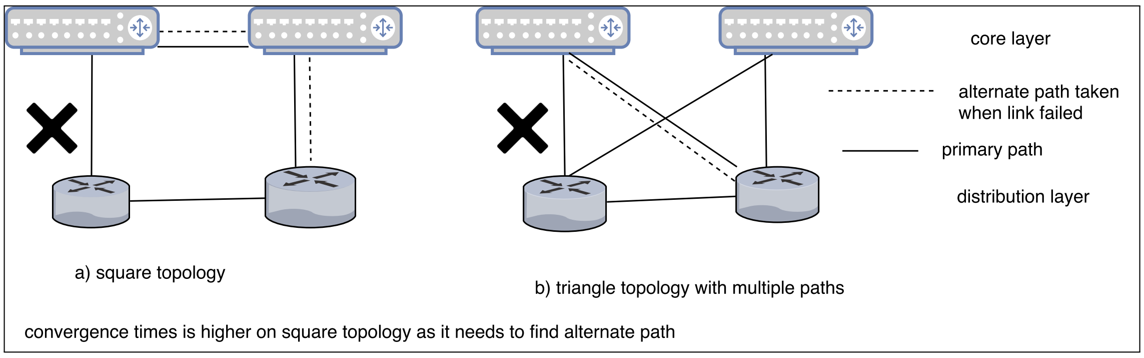

core layer
alternate path taken when link failed
primary path
distribution layer
a) square topology
b) triangle topology with multiple paths
convergence times is higher on square topology as it needs to find alternate path

Figure 2.1: square topology vs triangle topology

[1]OSPF and EIGRP are interior gateway routing protocols(IGRP).IGRP is used for routing within same domain

[2]time taken to find the alternate path when primary link goes down

- *Distribution layer*
  It is the middle layer between core and access network and sets up the boundary between layer 2 and layer 3. Layer 3 goes upstream to the Core and Layer 2 downstream to the Access layer. Another value aggregation layer provides is the gateway redundancy. Edge devices connected to Access layer can use multiple aggregation layer switch as their gateway. It is also called service layer since it provides service module integration. It provides service like load balancing, firewalls, Access control lists (ACL), security and traffic optimization. Spanning tree processing takes place at this layer.

- *Access layer*
  The Access layer is the lowermost layer where end devices and servers are connected. It consists of the switches that support both layer 2 and layer 3 topologies

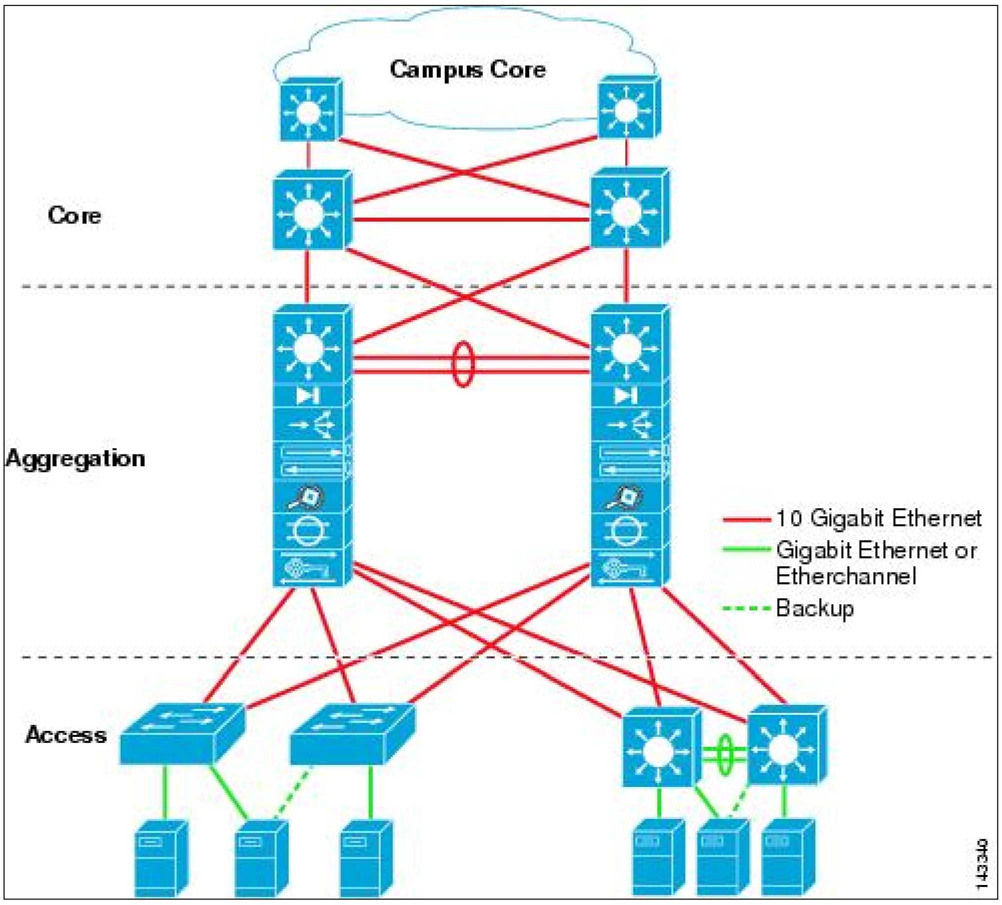


Figure 2.2: basic layered design[8]

2.1.2 Data center tier

Data center tier first came into the scene in 2005 as a quantifiable standard that can be measured against individual data centers. These standards were developed and maintained by TIA (Telecommunications Industry Association). Uptime Institute, 3rd party research institute developed 4 separate tiers. Although these two standards were developed and maintained by the different organization, they are very similar to each other in term of criteria and requirements on each tier[22].

Data center tier ranking primarily focus on data center's infrastructure, levels of redundancy and guaranteed level of uptime.

There are 4 tiers of data centers as[42][22]:

1. *Tier I*
   Tier I is simplest among the four tiers with almost no level of redundancy. It is composed of a single path of cooing, power distribution. It is expected to be available 99.671% of the time.

2. *Tier II*
   Tier II is composed of a single path for power and cooling distribution, with redundant components, providing 99.741% availability. It is less susceptible to unpredicted downtime than Tier 1 data center. These data centers will have some backup elements, such as a backup cooling system and/or a generator.

3. *Tier III*
   Tier III data center has multiple cooling and power path for all the equipment. If one path fails another takes over. This allows a data center to have a higher uptime.Maintenance/updates/upgrades can be done without downtime in this category of data centers.99.982% of time tier 3 data center is operational.

4. *Tier IV*
   Tier IV represents top-tier data center and represents data center having fault tolerant infrastructure. It is composed of multiple active power and cooling distribution path. This tier data center has redundancy in every equipment. Multiple cooling units, backup generators, power sources, chillers, etc. If one piece of equipment fails another takes place immediately without causing downtime. This tier of data centers have expected availability of 99.995%

2.1.3 Data center power consumption and Emission

Data centers currently consume about 3% of the global electricity production and contribute to 2% of the global greenhouse gas emission which is equal to what whole air travel industry contributes 416.2 terawatt-hours of electricity consumed by data centers in 2015 exceed the total power consumption of whole UK[4]. Most of the data center power consumption comes from the energy expenditure on cooling. Britain's leading data center expert Ian Bitter predicts power consumption of the data centers to grow by 3 folds in the coming decade [4]. Reducing the carbon footprint have been one of the larger priorities. Last 50 years or so has seen the record increase in surface temperature and prime culprit – the tremendous increase in greenhouse gas emission. Google achieved 15% reduction in power bill and 40% in cooling expenses with the use of Deepmind AI[34]. Some organizations like Facebook and Apple as a step to reduce carbon emission are building their new data center to entirely run on renewables[29],[40]. However, most of the data centers depend on the conventional grid for the electricity. Renewables contribute very little when it comes to global electricity production. Another thing to consider is that renewables plants are expensive than conventional one to build. However, it is expected to be cheaper in near future to construct and run with advancement in the storage (battery) and technology.

In the article[20] published by Stanford University, Mark Golden pointed that 88% of the carbon dioxide emission can be cut by increasing efficiency of IT infrastructure and combining with some smart power management choices such as building data center in Nordic countries where the temperature is very low. When combined with renewables it claimed that emission can be cut by staggering 98%.

2.2 Cloud computing

With an ever rising demand for more powerful and flexible computing resources, Cloud computing is on the rise. Cloud computing enables the business to move to internet from traditional in-house computing on the local machine. Cloud computing is the system where computing and storage resources are shared among various users. Virtualization of physical resources such as CPU, memory, and disks allows cloud computing to share the same physical resource among numerous user which otherwise would have been not possible. This, in turn, leads to a reduction of cost of operation, low or no maintenance overhead, flexible computing and scalable resources.

"cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [31]".

"Simply put, cloud computing is the delivery of computing services—servers, storage, databases, networking, software, analytics and more over the Internet ("the cloud")[1]. "

"cloud computing is a style of computing in which scalable and elastic IT-enabled capabilities

are delivered as a service to external customers using Internet technologies [18]."

"Cloud computing, often referred to as simply "the cloud", is the delivery of on-demand computing resources - everything from applications to data centers over the internet on a pay-for-use basis [23]."

We can draw some main characteristics of cloud computing from the definition above as:

1. Shared resources.
2. Scalable computing resources, User can increase or decrease resources as per their need or is dynamically allocated as per the use.
3. service accessible through the internet.
4. pay per use.

2.2.1 Cloud service delivery model

A Cloud service delivery model is a way of providing computation to users over the internet.There are three main category of deployment model. These models are compared in figure 2.3.

2.2.1.1 Infrastructure as service (IaaS)

Infrastructure as service offers the computing resources but in the virtual environment so that multiple users can access them. These resources include Data storage, servers, networking and virtualization on pay per use basis. Resources will be managed by service providers and a user will use these resources for running their applications and storing data. IaaS eliminates the need for having dedicated hardware. IaaS is mostly used by organizations who want computing infrastructure but does not want to purchase and maintain them. Amazon Ec2, GoGrid, rackspace.com, IBM SmartCloud Enterprise, Microsoft Azure etc. are some providers of IaaS

2.2.1.2 Platform as service (PaaS)

PaaS mainly consists of programing language execution environment, operating system, Web server and Database.It is the platform for developing, testing and managing software applications. Users can build, run and compile their program without worrying about the architecture underneath. User controls data and application resources and cloud vendor manages the rest. AWS Elastic Beanstalk, Google app engines, Redhat open shift, Engine yard are some examples.

2.2.1.3 Software as service (SaaS)

SaaS delivers the application via internet.Users can run the applications or software on the cloud via the internet and pay according to usage. The user does not need the application installed on their local PC to use these service. This model is platform independent since the service runs on the cloud, not the client computer. Service provider runs the single instance of software and is accessed by multiple users via a web browser or lightweight client applications. All the computing resources are managed by the vendor.This kind of service is used by end users. Microsoft 365 and google drive are among the most popular SaaS applications.

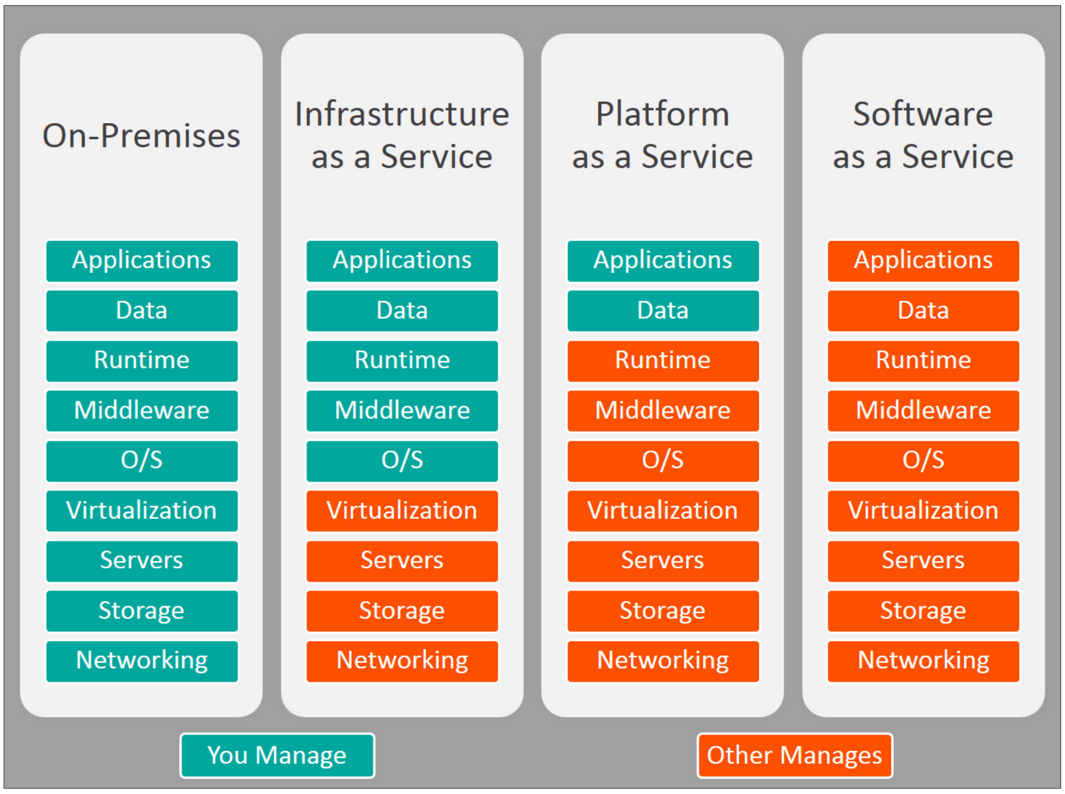


Figure 2.3: comparision between IaaS,PaaS and SaaS [44]

2.3 Cloud deployment model

The Cloud deployment model represents the specific cloud environment distinguished by ownership, access, and size. There are four broad categories of cloud deployment model

2.3.1 Private cloud

In private cloud, the organization owns the infrastructure and is set up within organization internal data center. These infrastructures are for the use of the single organization. Cloud vendor provides cloud services and resources and they are shared and used by the employees (or users) within that organization. Infrastructure is managed and operated by the organization or third party regardless whether it is on or off premise. Unlike the public cloud, organization have Control and is much secured because the infrastructure is not shared with others.

2.3.2 Public cloud

It is based on the standard cloud computing model in which resources such as storage, applications etc are made available to the general public via the internet. These services may be free or based on the pay-per-use model. Users access the service through web browsers or client application. Users share the resources and services among them. Public cloud service can be set up at low costs since the hardware, application and bandwidth costs are covered by the provider. The cloud service provider has the full control of the public cloud with its own policy, value, and profit, costing and charging model. Examples of public clouds include Amazon Elastic Compute Cloud (EC2), IBM's Bluemix Cloud, Sun Cloud, Google AppEngine and Windows Azure Services Platform.

2.3.3 Hybrid cloud

Hybrid cloud computing is the mix of both private cloud and public cloud that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability. It is basically private cloud linked to one or more external public cloud services. Organizations can optimize their resources by moving peripheral business onto public cloud while controlling core business on the private cloud which enhances their competencies.

2.3.4 Community cloud

Several organizations with common computing needs construct and share the infrastructure. It is managed either by within one of the participating organization or third party vendors. The resources can be shared for scalability and cost reduction yet maintain the security that a public cloud cannot offer.

2.4 Server monitoring tools

There are lots of server monitoring tools that provide information about the health of the server. There is broadly two classes of monitoring tools: agent-based and agentless. Table 2.1[3] provides the detailed comparison of these two approaches. Few popular server monitoring tools are listed below.[4]

1. Nagios
2. Icinga
3. OP5
4. Zabbix
5. Munin
6. Datadog
7. Copper Egg
8. Instrumental
9. Anturis

[3]https://www.quora.com/What-are-the-pros-and-cons-of-agentless-versus-agent-based-server-performance-monitoring

[4]Compiled from https://haydenjames.io/20-top-server-monitoring-application-performance-monitoring-apm-solutions/ and https://stackify.com/top-server-monitoring-tools/

	Agent based	**Agent less**
Definition	System requires agent to be installed on the system to be monitored. This agent does the task of collecting the data from the system under monitored.	Operations where no service, daemon or process (AKA an agent) needs to run in the background on the machine the action is being performed on
Data collection	Agent automatically collects the data and either sends periodically or when masters polls	some authentication mechanism is needed in the system being monitored (like user account or RSA fingerprint) The monitoring program collects the file or data directly from target machine via protocols like telnet, SSH or FTP
Network dependence	Agent can work even when there is temporarily lost of connection to monitoring station(master)	Always connection is required.
Expandability	Agents can be customized and expanded	No such capability
Overhead	Less bandwidth overhead as data is collected locally	Large bandwidth overhead as raw performance data is transported to a remote data collector.
Maintenance	requires occasional patching and troubleshooting of agents in target system. Difficult in case the target system are dispersed around large geographical location	No maintenance

Table 2.1: Agent based monitoring V/s agentless monitoring

2.5 IBM cloud

IBM cloud, formerly IBM Bluemix cloud is the set of computing services from IBM that provides both Platform as service (PaaS) and Infrastructure as service (IaaS).

IBM cloud is used throughout this thesis as IoT platform. IBM cloud is based on Cloud foundry open technology. It supports the programming languages like PHP, Java, Python, Node.js, Go etc. It provides powerful web dashboard that can be utilized to create, run, view and manage various applications and services

The IBM Internet of things foundation is powered by IBM's following leading products and services: IBM DataPower Gateway, IBM WebSphere Application Server Liberty Core, IBM Informix TimeSeries, IBM MessageSight, Cloudant, and SoftLayer[27].

2.5.1 IBM Watson IoT Platform

IBM Watson IoT platform is the IBM's internet of thing platform, built within the IBM cloud. It provides a means, by which the app in IBM cloud can access IoT devices and data that can be employed to create analytics applications and mobile IoT apps. Watson IoT platform is built on following major area.

Watson IoT platform allows powerful device management operations, store and access device data, connect a variety of devices and gateway devices. All the communication to and from devices and IBM cloud is encrypted and secured by using MQTT and TLS protocol.[12]Architecture of IBM Watson IoT platform is illustrated in 2.4.

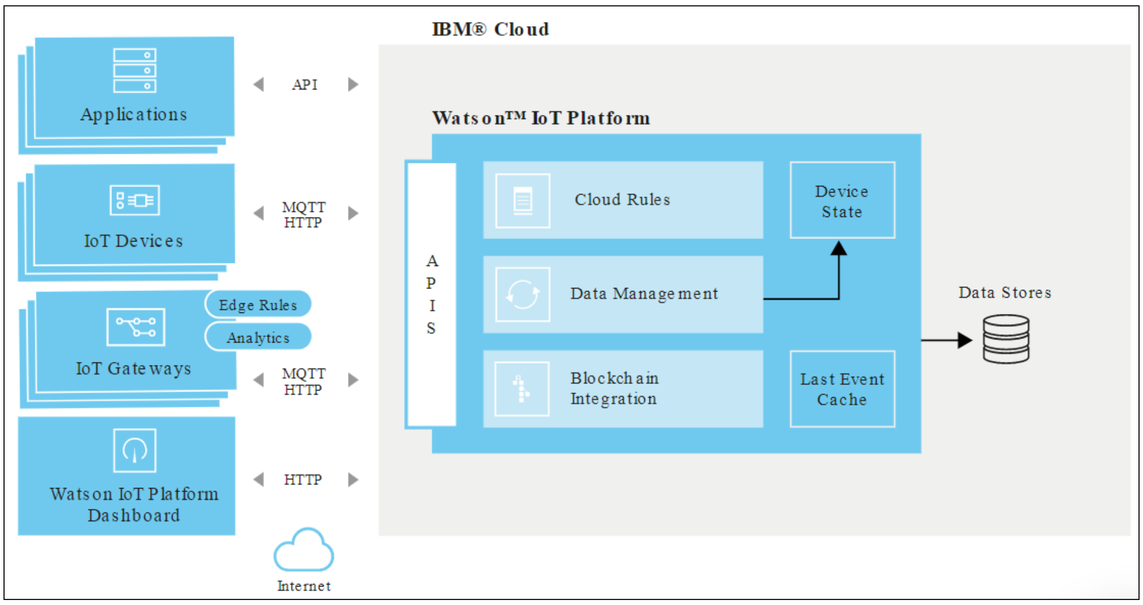


Figure 2.4: IBM Watson IoT platform architecture.[12]

Watson IoT platform API and the Watson IoT messaging protocol facilitates communication between device and application. Device data can be stored or used with analytics solution. Watson IoT platform dashboard as a front-end user interface provides a powerful web-based tool to manage device and data. Data can be visualized in a numerous way such as histogram, bar chart, pie diagram, line graph, donuts diagram etc. Watson IBM Dashboard provides a variety of other functions such as

Members: Add or remove members/users who have access.

Access management: Define roles for various users. It is similar to access control.

*Members:*Rules: Refers to the set of conditions that are triggered when certain criteria are met and action to take when rules are invoked. Rules are triggered when there is a change in state of a device. For eg. taking the action such as sending an alert to the system administrator when the temperature reaches a critical point.

*Members:*Security: Blacklisting or whitelisting specific IP or countries. The security level can also be set from here. To get the set up working security level is set to TLS optional.

*Members:*Extension: Extensions are optional service integrations which can be added to Watson IoT platform or integrate with the third-party service

2.6 Working mechanism of IBM cloud

The General representation of working of IBM cloud is represented in a figure 2.5. Raffaele Stefani in his Redbook [39] explains working of Bluemix(Now IBM cloud). Either it is a web app or a mobile app, the execution environment is different for each app, although those apps may reside on the same virtual machine (VM).

When developer deploys the app to cloud foundry, IBM cloud determines appropriate VM for the app to run on. Factors like load already on VM and runtimes or frameworks supported by VM are looked into while determining the VM. Application manager installs the appropriate framework and runtime and app can be deployed within that framework. In any VM, Application manager handles the communication with rest of IBM cloud infrastructure and manages apps deployed within that VM. Each VM holds the container that isolates app from other. IBM cloud installs framework and runtime within this container for that app.

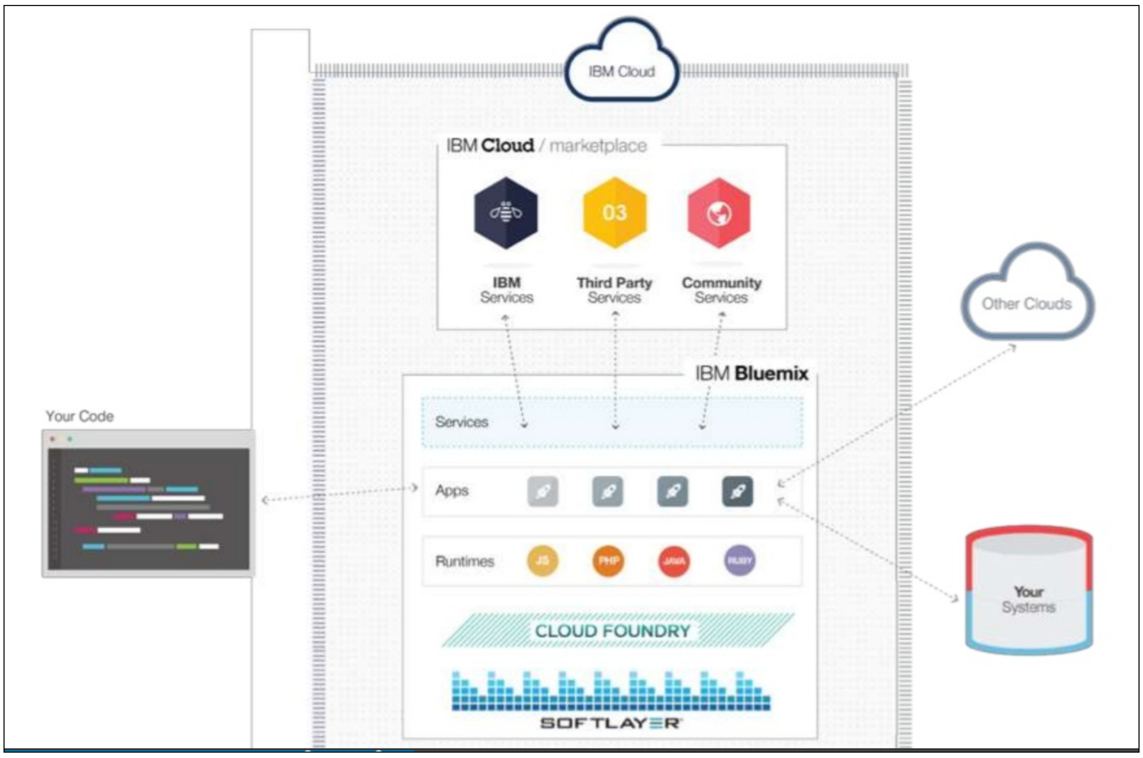


Figure 2.5: How does bluemix works.[17]

2.7 Cloud Foundry

Cloud Foundry(CF) is the open source platform as a service(PaaS) on which developer can build, deploy, run and scale applications [35].

CF is licensed under Apache 2.0 and supports Java, Node.js, GO, Python, PHP, Ruby, .NETcore and Staticfile. Use of CF eliminates developers to configure their app to run on the specific platform. Unlike traditional cloud apps which are bound to be deployed on vendor bounded cloud, CF provides a platform that can be used to deploy on a variety of IaaS like AWS(Amazon web services), vSphere, OpenStack, IBM cloud etc. This prevents potential vendor lock-in. CF allows developers to create an app in multiple frameworks or languages. It enables developers to use their own tools and code without concerning about the architecture the code is going to run. CF can be deployed on organization's own internal infrastructure (private cloud) or on cloud providers infrastructure like IBM cloud.

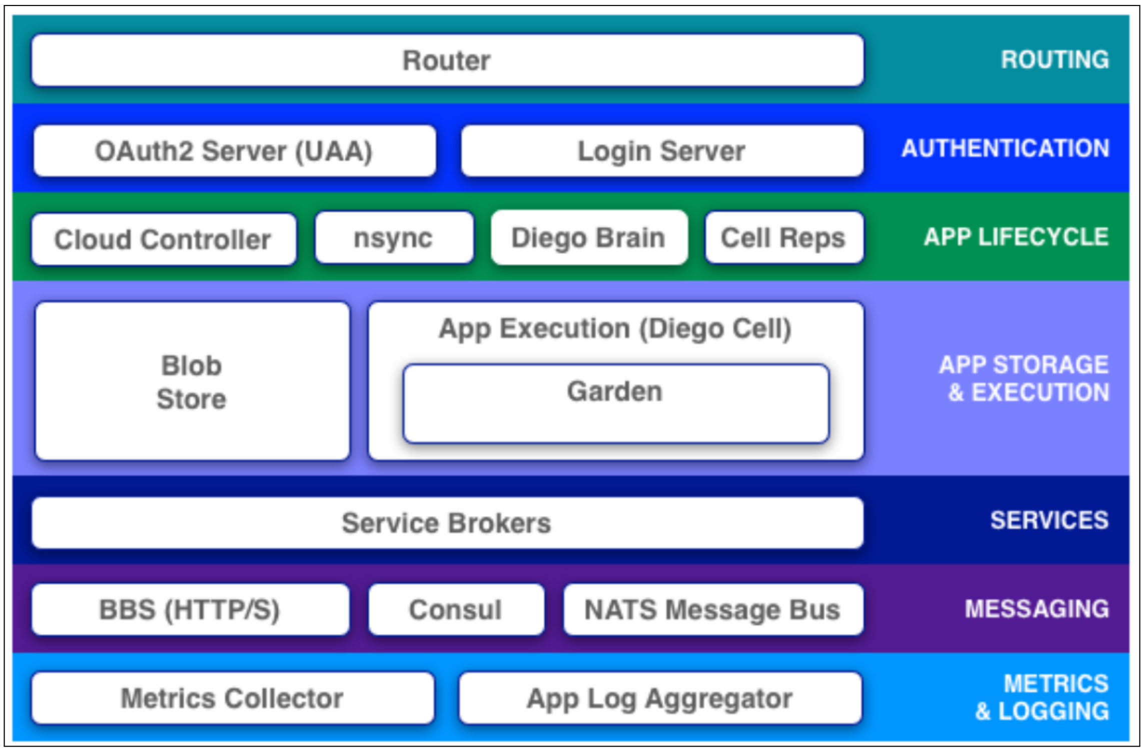


Figure 2.6: Architecture of Cloud Foundry[13]

2.8 MQTT

MQTT (Message Queuing Telemetry Transport) is a publish/subscribe, extremely lightweight and simple messaging protocol[26]. It supports machine to machine (M2M) communication and designed with low bandwidth energy constrained battery powered devices. MQTT user shorter header and packet size than HTTP. Each MQTT messages have fixed header of 2 bytes. MQTT uses publish/subscribe communication model. Client/sender publishes a message with some topic to the broker. A broker is an intermediary node or device that receives all the messages, filters it and publishes the message to the subscribed clients. The receiver receives the message with the topic it subscribed to. The illustration from Khagendra's Thesis[3] in figure 2.7 shows working of MQTT protocol.MQTT is used in this thesis to send data to IBM cloud

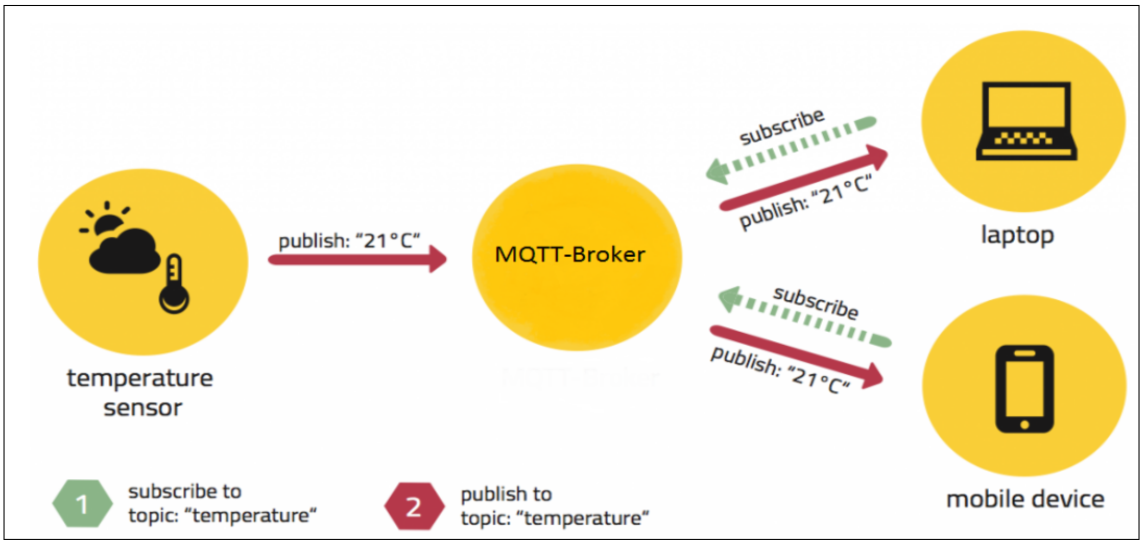


Figure 2.7: MQTT publish/subscribe architecture

2.9 NUMA

NUMA is the computer memory design used in multiprocessing or parallel processing architecture. It is the method of improving performance and expanding the ability of the system by clustering the microprocessor to share the memory locally. In this design, each processor has its own local memory but can access the memory which is not local to it. NUMA is used in Symmetric Multiprocessing(SMP). In SMP architecture, multiple processors are running under the single Operating system(OS) and use common memory and bus or interconnect path. As more processors get added, the shared bus will be crowded and performance will degrade as the result. NUMA groups the number of processors into a cluster called NUMA node. The Number of processors per NUMA node may vary with the hardware vendors. Each NUMA node will have their local memory which all node member can share. This eliminates the need to access crowded. One trade-off of NUMA design is that it will take longer to access some part of memory(Memory which is not local to NUMA node). ALTO and research 5 test server both have 8 NUMA node with 8 CPUs in each node. The figure 2.8 is a schematic illustration of NUMA implementation in ALTO.

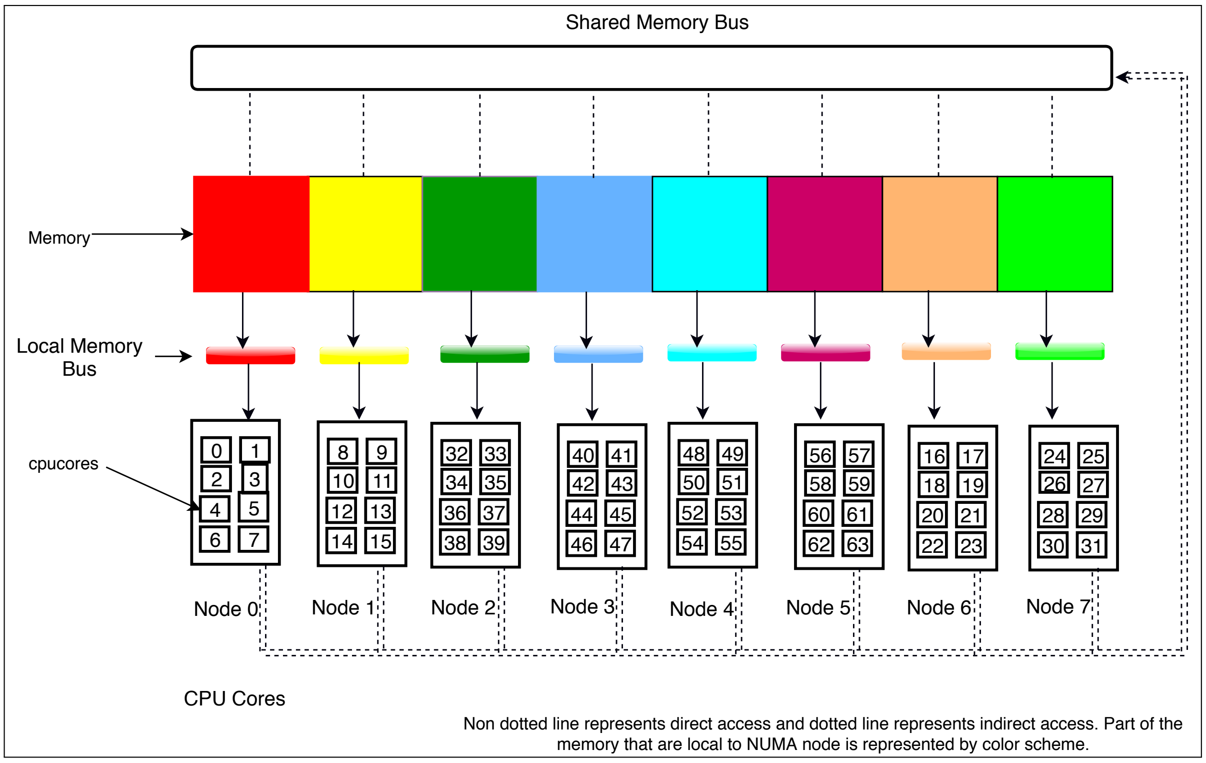


Figure 2.8: Schematic representation of NUMA node in ALTO and Research5

2.10 Node-Red

Node-Red is the flow-based programming tool originally developed by IBM. It is the visual tools that let connection between different components such as device, the database just by wiring them together. It is a drag and drop technique and quickly help to create flow between different services or devices. In this thesis, Node-Red is used to perform wiring between the device and the database so that device data can be stored in the database.

2.11 Machine Learning

Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate in predicting outcomes without being explicitly programmed. The basic premise of machine learning is to build algorithms that can receive input data and use statistical analysis to predict an output value within an acceptable range [36].

Field of study that gives computers the ability to learn without being explicitly programmed. - Arthur Samuel[5]

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E. - Tom Mitchell 5

The term machine learning was first introduced by Arthur Samuel in 1959 while working for IBM. In the 1950s he wrote the checker playing program. It was programmed to play repeatedly against itself. The program learns every time it plays and plays better. It learns from what are the moves that lead to winning and losing [28]. If we fit this classic example in Tom Mitchell's definition experience E is the number of time program plays, task T is the task of playing, performance P is chances of winning next game. Changes of winning improve more the game it plays.

2.11.1 Machine learning methods

Machine learning method is different techniques on how machine learning is used. There are different methods which are described below.

2.11.1.1 Supervised Learning

In supervised learning, the model is trained with labeled output i.e inputs data and their resulting outputs. The model predicts the result based on the data and their corresponding output it is trained with. The machine learning algorithm is provided with a set of inputs, along with their correct outputs. Then algorithm learns by comparing its actual output with correct output to find errors[10]. The input data are often termed as features and their output as labels. This kind of machine learning techniques is used when there is known data for the output which is being predicted. Supervised learning uses classification and regression technique.

2.11.1.2 Unsupervised Learning

In unsupervised learning, unlabeled data is provided to model and algorithm itself has to find some sort of pattern within the data. Machine learning algorithm has to derive some sort of relationship based on the input. Since data is unlabeled, we can not find the accuracy of a model like in supervised learning. Clustering is the most common algorithm used for unsupervised learning. Clustering algorithm groups

[5]definition extracted from https://en.wikipedia.org/wiki/Machine_learning accessed on April 6 2018

various data point into a number of clusters based on a relationship derived from input data. For example, it can analyze the shopping patterns of the customers and group them according to common interest[10].

2.11.1.3 Semi-supervised learning

Semi-supervised learning is used for the same applications as supervised learning. It uses both labeled and unlabeled data. A small amount of data is labeled and rest is unlabeled. The semi-supervised technique is used when the cost of labeling is high.

2.11.1.4 Reinforced learning

In Reinforced learning, learning agent learns from close interaction with the environment that is outside the control of the agent. This type of learning has 3 primary components: the agent (learner or decision maker), environment (everything agent interaction with) and the action(what agent can do)[10]. The agent takes some action on the environment. There is a number of action and agent does not know which action is best for given scenario. The agent performs an action and receives observation and reward from the environment. Observation can be state of the environment or additional information. Most of the time agent receives delay reward. In classic tic-tac-toe, program agent receives if the move was right only after the game is played. The goal of the agent is to maximize rewards over time.

2.11.2 Machine learning techniques

The machine learning techniques can be broadly classified into 3 major class

- Classification
- Regression
- Clustering

2.11.2.1 Classification

In machine learning, Classification is the technique that predicts a class or category y for the set of inputs x based on the training data[43]. Classification techniques predict discrete responses[6]—for example, whether an email is genuine or spam, or whether a tumor is cancerous or benign[25]. Support vector machine (SVM), boosted and bagged decision trees, k-nearest neighbor, Naïve Bayes, discriminant analysis, logistic regression, and neural networks are the common algorithm that uses classification.

2.11.2.2 Regression

Regression technique predicts continuous response [6], for examples change in power consumptions based on operating temperature. It tries to predict a continuous value for the inputs based on the previous information. It is in a sense similar to classification as both attempt to estimate a function that maps input to output based on early observations. But regression tries to map actual value not just the class of output.

2.11.2.3 Clustering

Clustering groups the inputs into a cluster that is similar or has similar characteristics. It does not need any kind of labeled data. For example, the algorithm might be supplied with photographs of animals and it groups animals into a number of clusters based on similar characteristics.

2.11.3 Machine Learning Algorithms

This section covers popular machine learning algorithms and those used in this thesis

2.11.3.1 Support vector machine

Support vector machines (SVM) are a set of supervised learning methods used for classification, regression and outlier detection. SVM was invented by Vladimir N. Vapnik and Alexy ya. Chervonekis in 1963.

Inputs to the SVM are the labeled data, with each output labeled as belonging to one or more category. SVM training algorithm builds the model that assigns new values to one category or others, making it a non-probabilistic linear classifier[45].

SVM attempts to find the best splitting boundary between data. The separating hyperplane is the hyperplane with the widest margin between support vectors[30]. It

[6] discrete value takes one value from set of all possible value. Continuous value can take any values

is illustrated in figure 2.9. In this thesis, we are using SVM with 2 different kernels linear and rbf(radial basis function). Kernel function in SVM is the representation of the similarity. It takes 2 inputs and finds how similar they are. The kernel as explained by instructors of *udacity* is a mechanism by which domain knowledge is injected into SVM algorithms.[7]. When the data is not linearly separable in the current dimension, the solution can be to map the data into a new dimension. Kernel achieves this. Very simple but effective explanation is presented in [9].

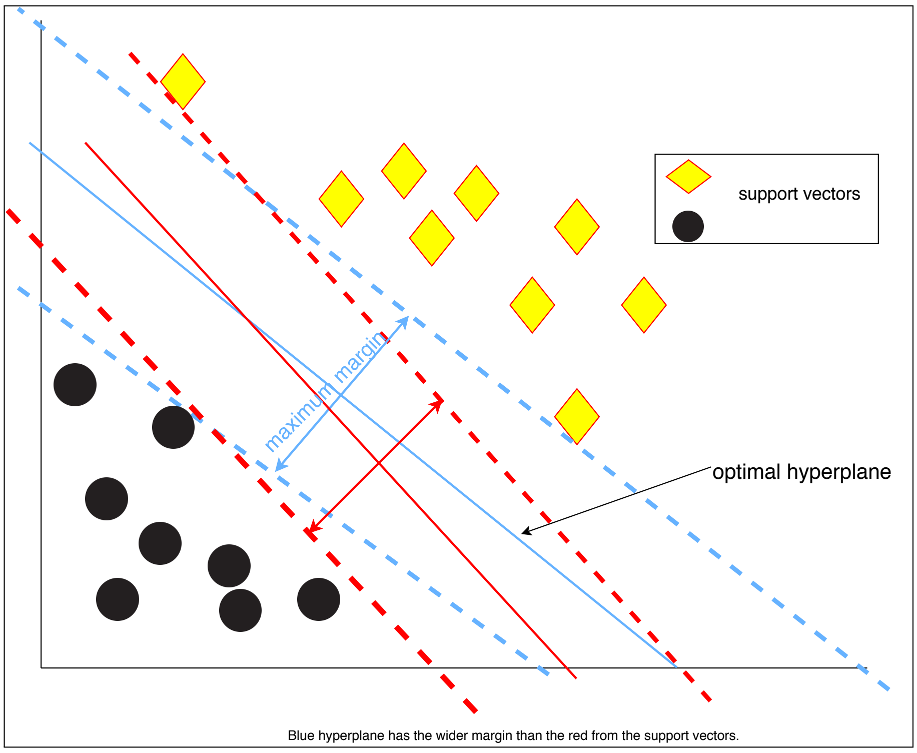

support vectors
maximum margin
optimal hyperplane
Blue hyperplane has the wider margin than the red from the support vectors.

Figure 2.9: SVM illustration of widest margin between support vectors

2.11.3.2 Artificial Neural Network

Artificial Neural Networks (ANNs) are inspired by the functioning of the brain. It tries to mimic the functioning of the brain which is intended to replicate the way human learn.

The main idea behind ANN is to mimic neuron[8], it consists of dendrites, a nucleus, axon, and terminal axon. Neurons transmit information via synapse between the

[7]https://www.youtube.com/watch?v=Ov_znn3KHfE
[8]neuron example https://youtu.be/oYbVFhK_olY?list=PLQVvvaa0QuDfKTOs3Keq_kaG2P55YRn5v

dendrites of the first neuron to the terminal axon of a second and so on. Neuron listens to this information from synapse connecting it and processes that information using its own function (which is, of course, complex and nonlinear).Every time the information is transmitted between the neurons, it is processed into the information that is more useful than the previous one. ANN is similar in the sense that it consists of the input layer, output layer and in most case hidden layers that transform the input into something that output layer can use[16]. Figure 2.10 illustrates the simple neural network.

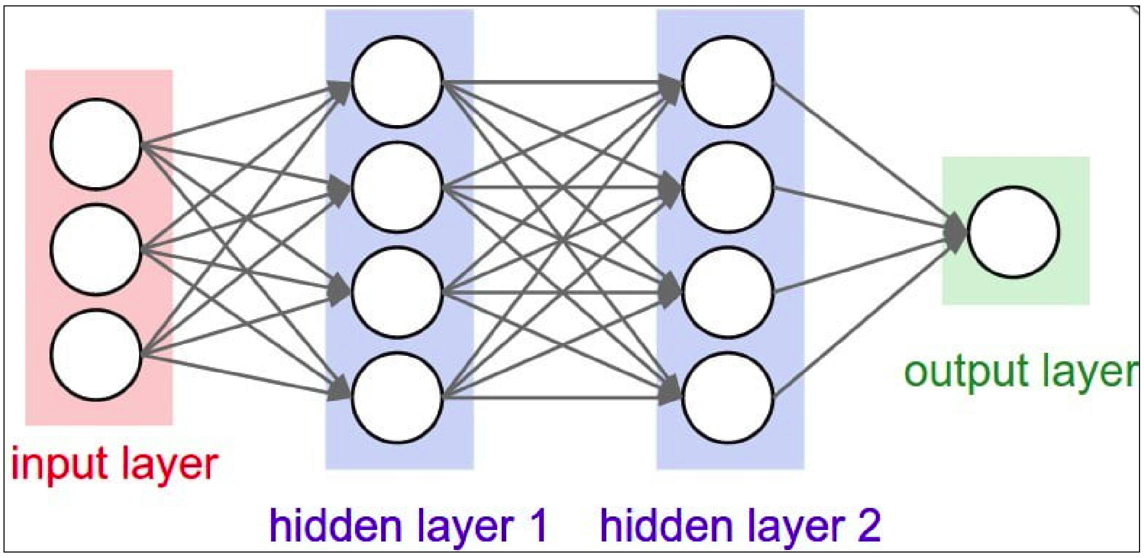


Figure 2.10: Artificial Neural Network[16]

2.11.3.3 Decision Tree

Decision trees are simple but yet effective algorithm for predictive machine learning. The main purpose of using decision tree is to create a training model which can use to predict class or value of target variables by learning decision rules inferred from prior data(training data) [37]. It uses a tree-like structure with root, branch and leaf nodes, with leaf nodes being one or more of the decision. Non-leaf nodes represent the attributes and leaf node represents a decision or prediction. The root node is the set of attributes in the dataset. The decision is made by walking through all the way to the leaf from root.

2.11.3.4 Random Forest

Random forest, as the name suggests it builds the forest. Forests are composed of a number of decision trees. Random forest builds multiple decision trees and merges them together to get more accurate and stable prediction [15]. More the tree in the forest more accurate the result is. The article [33] provides simple pseudocode for random forest algorithm. The pseudocode has 2 parts first is random forest creation and the second is a random forest prediction.

Pseudocode for random forest creation

1. Randomly select "k" features from total "m" features where k<m
2. Among the k features calculate the node "d" using best split point
3. Split the node into daughter nodes using the best split
4. Repeat step 1 to 3 until "l" number of nodes have been reached
5. Build forest by repeating steps 1 to 4 for “n” number times to create “n” number of trees

Pseudo code for random forest prediction

1. Random forest algorithm takes the test features and use the rules of each randomly created decision tree to predict the outcome and stores the predicted outcome (target)
2. Calculate the votes for each predicted target
3. Prediction with the highest vote is considered as the final outcome.

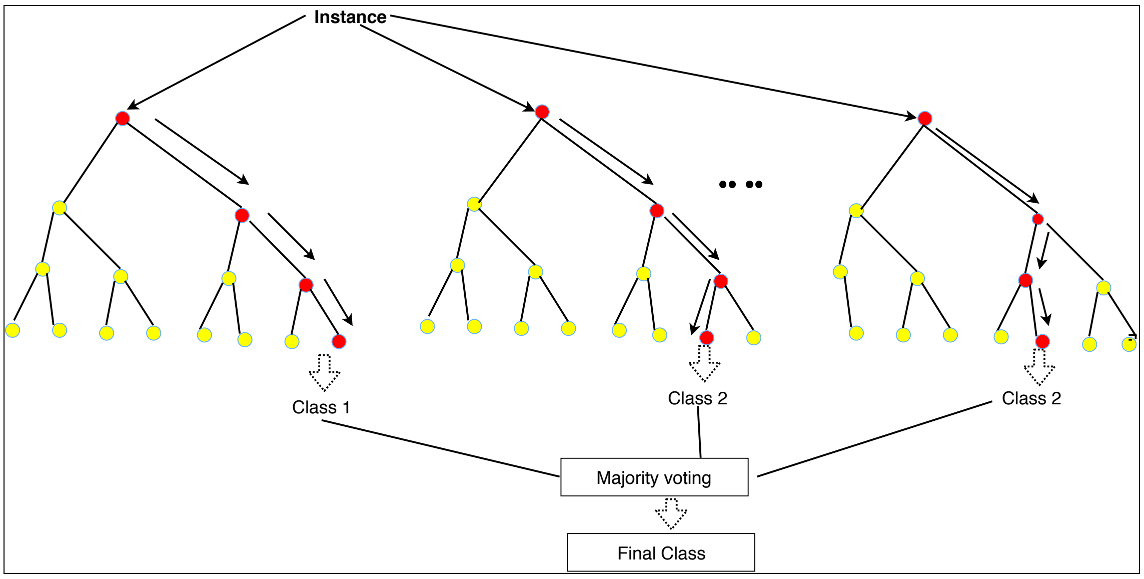


Figure 2.11: Random forest illustration[24]

2.11.3.5 Linear Regression

Linear regression is probably most basic algorithm in the machine learning. Linear regression attempts to find a best-fit line for the dataset that minimizes the mean square errors. It achieves this by solving the classic equation of straight line y=mx+c. where m is the slope and c is y-intercept with x being the input and y being the output that is to be predicted. Linear regression performs better when there is a higher correlation between the data.

2.12 Tools

This section explains the basic tools used throughout the thesis

2.12.1 Python

Python is one of the most popular scripting language. Simplicity and preciseness make python almost default choice. The script/program can be written with much less code than other languages like C/C++, Java etc. Python has very good documentation, great forums, and large user base. Python provides a variety of library and packages that make programming even simpler.

2.12.2 R

R is the programming language and free software environment for statistical computing and graphics. R is basically used in correlation test during this thesis. It provides excellent analysis tools.

2.12.3 Scikit learn

Scikit learn is the machine learning library for python. It provides a number of supervised and unsupervised machine learning algorithm on the go. Scikit learn is build upon NumPy, SciPy, and matplotlib. It can be used for regression, classification, and clustering.

2.13 Related works

Lots of works had been conducted to reduce the energy costs of the server and also to discover the relationship between the efficiency of the devices and operating temperatures some of which are explained below.

In El-Sayed et al [38] did the comprehensive study on the effect of temperature on the hardware reliability. They traced that higher temperature does not have a significant impact on LSE(latent sector errors) than that in average temperatures. Their experiment showed that disk performance begins to show observable degradation at an ambient temperature of over 40-degree Celsius. However, data centers are rarely likely to have operating temperature above 35. Increase in the power consumption can be attributed to increasing in fan speed and CPU power leakage is negligible. Observations reveal constant consumption for ambient temperature till 30 degrees Celcius and continually increases until 40-degree Celsius and increase is quite dramatic up to 50 %. Another interesting observation is power usage starts increasing for the same temperature despite the workloads. Effect of temperature on system reliability is much less than often assumed. Data centers can reduce carbon emissions and power consumption if threshold operating temperature is increased by few degrees. But one should not only take account of average temperature but the temperature of hotspots, which is the hottest part of the data centers. One of the most important sightings of this work was variability of the temperature contributed high to the failures than higher temperatures. So variation in temperature can be one of the factors causing failure rather than average temperature. Meaning that data center should focus on maintaining the consistent temperature to save hardware from failures than prioritizing to run at low temperature.

In Pinheiro et al [32] collected data from detailed observation from large disk population in a production Internet services deployment and provided statistics of failures and correlated failures with various parameters. They found nonuniform failure patterns among 3 different time period of observation when disk utilization was taken as a factor. Within first-year high utilization disk have a higher annual failure rate(AFR) but 3 years group have opposite results and again 5 years group indicate high utilization disks are more prone to failure. The contribution of higher temperature on failure is significant only for the high range of temperature(above 45 C) and among the aging disks(lifetime over 3 years). Similarly, disks with scan errors and Reallocation counts greater than 0 are highly probable to failure without them.

In harper et al [6]purposed a thermal-aware workload scheduling algorithm (TASA) for a homogenous datacenter which schedules the *"hot"* job (tasks which are going to increase a temperature of the node much higher than others) in the cold compute nodes. They defined Compute resource model as a function of space, time and temperature and workload model based on resources required, time and temperature. The algorithm also defines the Online task temperature calculation model which calculates predicts the temperature of compute node and ambient temperature as the result of running the job. Input jobs to the algorithms were real job based on the logs of Center for Computational research center (CRC) for the period of 30 months. TASA achieved 6.1-degree Fahrenheit of max temperature reduction in CRC when compared to first come first serve (FCFS) scheduling algorithm used in CRC. But the reduction in temperature was achieved at cost of increased job response time by 13.9

In L.N bairavasundaram et al [2] analyzed the data collected from production storage systems across 1.53 million disks over 32 months. The study encompassed both nearline and enterprise class disks. the study revealed that enterprise-class disks are less prone to LSE compared to nearline disks. Disk with at least an error is more likely to develop additional errors than disk without errors. Enterprise disks are as likely to develop additional error as nearline disks once it develops at least an error. Odds of developing LSE is higher in higher capacity disks. According to this investigation, more than 60% of LSE can be detected through disk scrubbing.

Khagendra Basnet in his thesis [3] did a detailed stress test on Research5 Test Server and ALTO server. He stressed all NUMA nodes one after another in research5. He observed that particular temperature sensors were more sensitive to stress particular NUMA node. For eg., while stressing NUMA node 0 temperature sensor 1 was most sensitive as it showed a higher variation of temperature. Behavior was similar for NUMA node 2 with temperature sensor 3. A similar experiment was run with ALTO server, but there was no clear result like that in Test Server where particular temperature sensor could be related to particular NUMA node. As his another contribution, he picked out best and worse CPUs in both Research5 and ALTO server. Best CPU are the ones which relatively maintained a low temperature during the stress test and worse CPUs are the ones which when stressed, temperature sensors recorded at a higher temperature. When stressing Best and Worst CPUs and comparing their thermal performance in Research 5, test server with no external loads he found out there was a difference of almost 5 degrees between the peak temperatures. But the difference was just around 2 degrees when the experiment was run at production server ALTO.

Part II

The project

Chapter 3

Design

This chapter cover the how the project is planned.After the completion of design we will have clear road map on how our plan is will implemented into action.

3.1 Design consideration

Various parts of the projects such as monitoring platform, data collection methods etc. were figured out before actually implementing at the project. This chapter covers several of these aspects.

3.1.1 Monitoring Platform

The first step in deciding how to start the project was to figure out how to develop monitoring platform. There were two primary alternatives.

- **Develop a monitoring system over external cloud** The data (from ALTO/Research5) will be sent periodically to the cloud. Those data will be visualized over the cloud. We may use plugins to visualize data or use the visualization tools provided within the cloud. Developing the monitoring app for the cloud environment was not an option given a short duration of this thesis.
- **Build a monitoring system** We were provided with test server Research5 for this project. Research5 and ALTO are explained in more detail in section 3.2. We can use Research5 to collect data from ALTO and visualize it. Of course, we will have to first build monitoring system on Research5. ELK (Elastic search Logstash and Kibana) stack would have been a top choice for the

monitoring system in Research5. ELK generates graphs and plot and provides nice visualization through a webpage. However, ELK utilizes logs to generate those fancy graph. Since we were interested to see if there is any relation between power consumption and internal temperature. We need the way to extract power and temperature readings along with system resource utilization periodically. We were not sure if we can get the periodic temperature and power reading with ELK.

Another alternative to ELK was to build custom monitoring tool to periodically plot the readings from the system being monitored. This is a costly job since everything for the custom monitoring system had to be created from scratch. Considering short time frame for this thesis (17 weeks), this did not look like a viable option. Also, non-root user access was provided on research 5 test server so anything requiring root access was not an option.

So our chosen alternative was to push the readings from research5/ALTO to the cloud and use the tools and plugins provided there. There were various options like IBM cloud, Amazon AWS, Google cloud platform. IBM cloud was chosen as the monitoring platform because they provide 6 months free access to the student of IBM Academy (both Oslomet(formerly HiOA) and UiO are a member of IBM Academy). Another key reason for choosing IBM was to take advantage of the setup from Khagendra's previous work [3] where he built an IoT platform and the same platform can be reused with minor adjustments.

3.1.2 What data to collect

Next step was to determine what values we want to collect. During the early stages of the project, we were more interested in researching the cause of failure for hard drives in ALTO because one of the major cause of downtime in ALTO was disk failures. But hard drives in ALTO were not SMART enabled. SMART data indicates a possible imminent drive failure. However, SMART enabled disk do not always catch imminent failures. We were also unsure whether ALTO has the historical record of failures with diagnostics. So we shifted our goal from the hard drive failure investigation. Our new interest was to investigate the relationship between temperature and power consumption. So power and temperature readings need to be collected. We were also building simple monitoring system so we need to collect values relating to CPU, memory and disk utilization. Table 3.1 summarizes the values we are going to collect during this project.

Temperature	8 temperature readings extracted from output of sensor command
Power	4 power readings extracted from output of sensor command
CPU	64 reading of individual CPU core utilization and 1 Total CPU utilization compiled from output of /proc/stat command
Memory	Total memory, Memory used, Free memory, Swap memory and swap used extracted from output of /proc/meminfo command
Disks	Disk size , Disk used and available space from output of df command

Table 3.1: Summary of the various values being collected during this project

3.1.3 How to collect data for analysis

Next design consideration was on how to collect the data for analysis. Data was to be collected every minute for over a week from ALTO server. There are two primary approaches. First is to append the value in the local CSV files which is later used during analysis. The second approach was to write the values extracted by the script into some external server. In this approach values from ALTO was to be written in the CSV files in Research5. One of the issues with this method was that there needs to be the exchange of RSA keys between the ALTO node running the script and research5. Since the script was to be run as superuser in ALTO exchange of keys was a bit risky and this approach was rejected for security reasons. The first approach also has a slight issue, since it was going to write on the file locally, file size was going to increase every time it writes, but since the script only writes numeric and string values, the file size for the data collection period for 7-10 days was going to be just a few megabytes at maximum. Hence the first alternative was chosen over the second one.

3.1.4 Stress Test vs No Stress Test

Previous work like Khagendra's thesis [3] used Stress Test on ALTO to generate CPU load, in order to increase the internal temperatures. Since ALTO is the production server, only maximum 10 minutes of stress test at a time was permitted during the period when there was low load in servers. In Research5 test server too, I was asked to stress for short period of time(10 minutes) if needed by the administrator. Although it was the test server administrator wanted to avoid any possible hardware failure. According to an analysis of previous works [3], [21], stress test for very short time may not cause an expected change in temperature or the change might occur some time after the test and may have gone unnoticed. If high CPU load from the stress test is going to increase the temperature, then stress test may not be necessary at all. ALTO is a production server. A number of students, faculty members, and researchers are using ALTO. There should be a period when there is high CPU utilization and period when there is a low load on the server. So we decided to collect data from ALTO every minute for the at least 7 days, which should provide us with enough data. So our approach will be to collect data from ALTO for an extended period without running any kind of experiment. Unlike stress test, we might not get the scenario where there is 100 percent CPU utilization. But we believe with our approach, we will get the data with greater variation. One advantage of this approach is that we are gathering real data instead of the simulated one. The result we get from the data from ALTO will represent real-world data. In case of Research5, we have to stress CPUs to see the effect since there is no real load in it.

3.1.5 Scripting Option

Python was the scripting option of choice because of various factors such as ease of use, wide range of library support and excellent documentation. R was used to generate the plots and graphs and correlation analysis. We could have used python for generating plots and correlation analysis. But R was chosen because of greater familiarity with it

3.1.6 Machine learning

Machine learning model developed during this project was to predict total power consumption. We are going to use five different supervised machine learning algorithm. We will be working with labeled data. The input data to machine learning algorithm will consist of 8 temperatures as features and a label as Total power consumption which is a sum of 4 different power readings. We are investigating how precisely machine learning model can predict the power. We are calculating mean square error and r2 score (also called as Coefficient of Determination). Mean square error is the measure of How close or far the predicted value lies from the estimator(actual output), the values close to 0 indicates there is less error between predicted and the observed output. The R2 score is the measure of how nicely data fitted into machine learning model. If results from our machine learning are promising then, Thermal aware load placement mechanism may be built.

3.1.7 Correlation Test

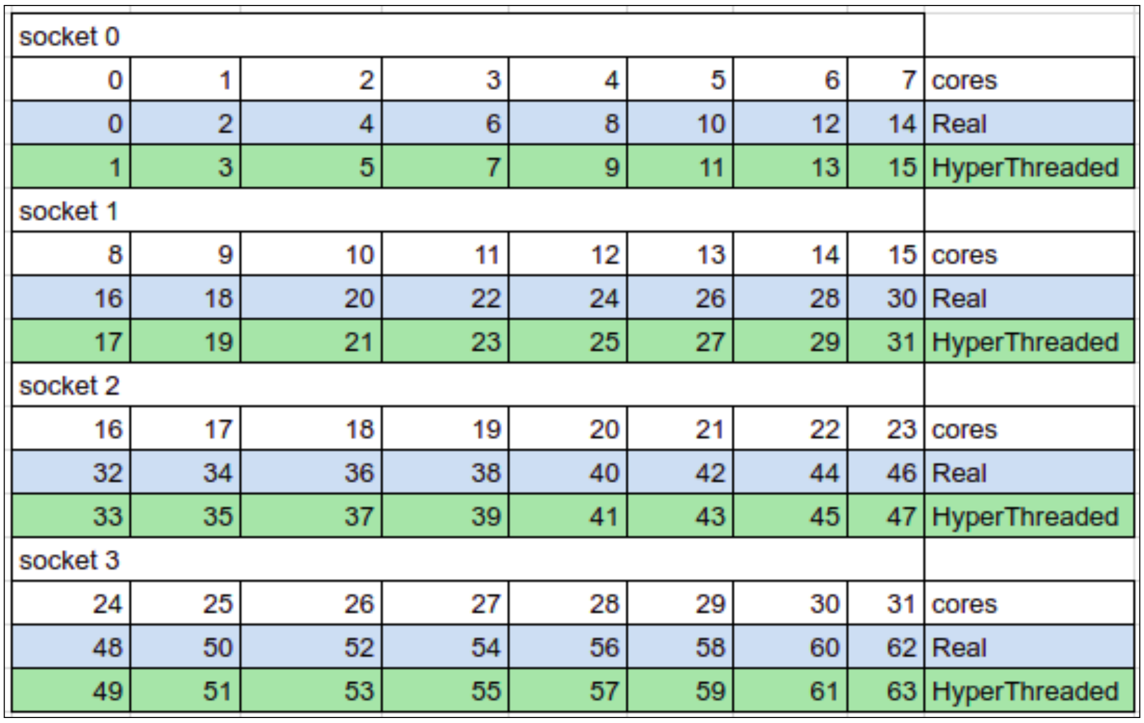
We will be performing Correlation test to find out how two variables are related. We will be performing correlation test between temperature vs power, CPU load vs temperature and CPU load vs power. We will be calculating correlation score also called as a coefficient of correlation. It has a value of +1 to -1 The result we get from correlation will reflect the effect of one variable on other. The perfect score of 1 or -1 relates that there is a linear relationship between two variables for which correlation was calculated. The value of 0 means there is no relationship. A positive correlation means an increase in one variable will increase the next one.

3.2 Test server vs ALTO cloud

Research 5, which is the test server used for scientific calculations and research purposes is used at the first phase of this project. It was the dedicated server with very less utilization. This server shares the similar CPU architecture with ALTO cloud's server. Research 5 was primarily used to develop the python script that extracts data from the system. The extracted data was sent to IBM Bluemix IoT platform for visualization. The script had to be tested and perfected at Research 5 before running at ALTO cloud.

ALTO cloud, on the other hand, is the production server with lots of students, faculty member and researchers running their projects and research. ALTO cloud was in use for the second phase of Project. Administrator of ALTO cloud is responsible for running the script.

Both Research5 and ALTO cloud server have same CPU architecture with 4 sockets each with octa-core design along with hyper-threaded enabled. Each socket with 8 cores means it has 4*8=32 physical cores in total and 64 logical cores. The architecture of the CPU is illustrated in figure 3.1 and specification of Research5 is represented in table 3.2. Both Research5 and ALTO have 8 inboard temperature sensors and 4 power sensors. We will attempt finding relations from collected as readings of these sensors.



socket 0								
0	1	2	3	4	5	6	7	cores
0	2	4	6	8	10	12	14	Real
1	3	5	7	9	11	13	15	HyperThreaded
socket 1								
8	9	10	11	12	13	14	15	cores
16	18	20	22	24	26	28	30	Real
17	19	21	23	25	27	29	31	HyperThreaded
socket 2								
16	17	18	19	20	21	22	23	cores
32	34	36	38	40	42	44	46	Real
33	35	37	39	41	43	45	47	HyperThreaded
socket 3								
24	25	26	27	28	29	30	31	cores
48	50	52	54	56	58	60	62	Real
49	51	53	55	57	59	61	63	HyperThreaded

Figure 3.1: Schematic representation of CPU architecture in alto and research 5

Table 3.2: Specification of Research 5 test server

name	AMD Opteron(tm) Processor 6366HE
CPU(s)	64
Socket(s)	4
Cores per socket	8
Threads per core	2
CPU MHz	1800.201

3.3 Final Design

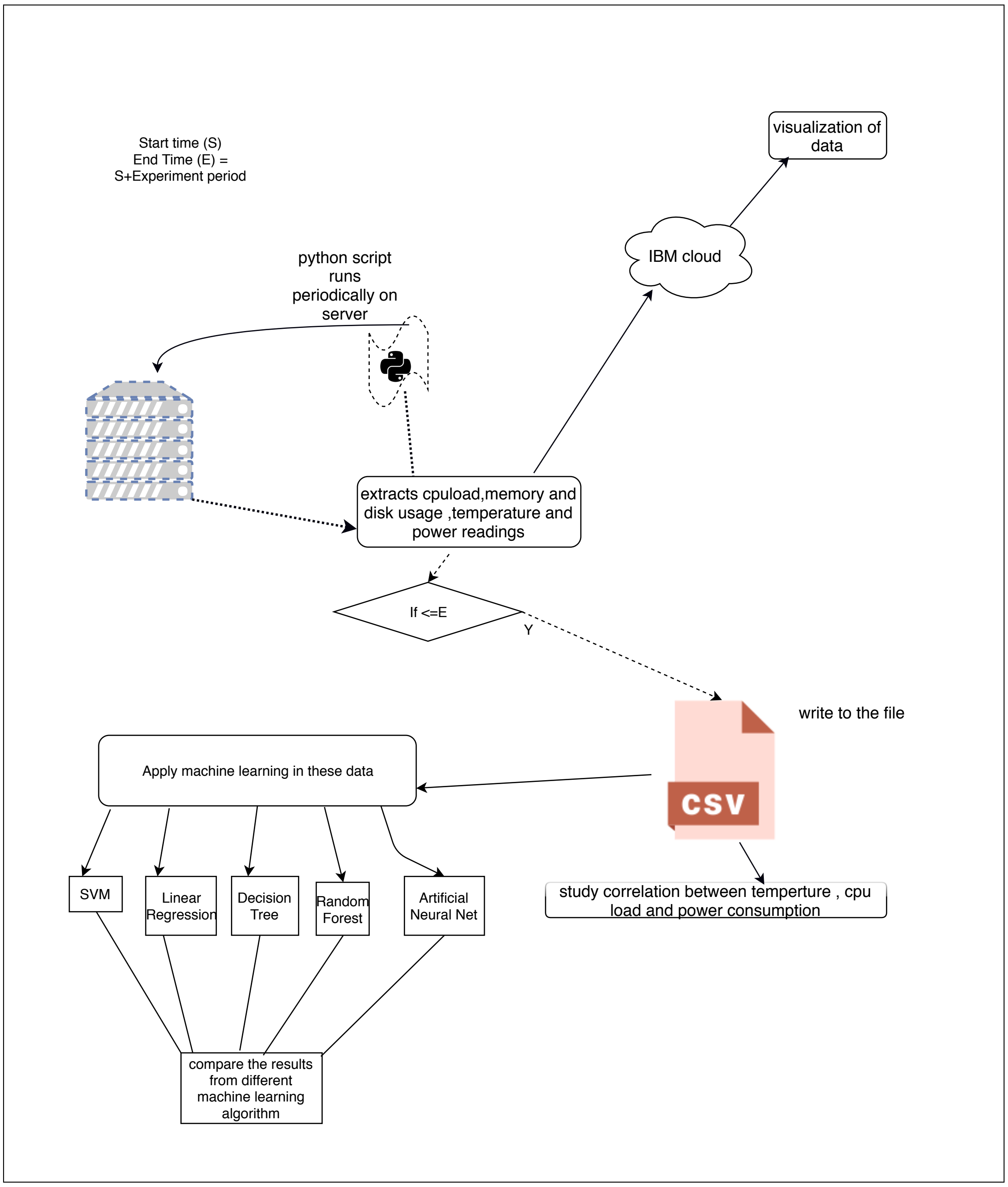

Start time (S)
End Time (E) = S+Experiment period
visualization of data
IBM cloud
python script runs periodically on server
extracts cpuload,memory and disk usage ,temperature and power readings
If <=E
Y
write to the file
CSV
Apply machine learning in these data
study correlation between temperture , cpu load and power consumption
SVM
Linear Regression
Decision Tree
Random Forest
Artificial Neural Net
compare the results from different machine learning algorithm

Figure 3.2: Overall Project design

Figure 3.2 shows the overall design of the project. Python script running on ALTO extract the data we need. It will run every minute as the cron job. The script will send the data to IBM cloud using MQTT protocol. Data will be sent in JSON format to IBM cloud. Visualization tools in IBM cloud will be used to visually represent the data so that system administrator will be able to see the status of the system. If a necessary alert system will also be developed using tools provided by IBM cloud. Alert system will send a message or email to Sysadmin if the values being monitored exceeds a predefined threshold. The script will also append the same values to local CSV files. We will only be collecting data for a week. After the week the cron job will be killed because of writing data infinitely into a file will increase the size and takes more disk space in ALTO which is a production server. The trimmed version of the script will run after a week which will just be sending the data to IBM cloud. The CSV files from ALTO will be used to perform the correlation test. Data from the same file will be used for machine learning. We are primarily going to perform machine learning primarily with SVM and Artificial Neural Network. Other machine learning algorithms also may be tried depending upon how much time is left for analysis. We will be comparing the accuracy of the various machine learning approach. The script will be first tested in Research5. Data collected from Research5 also will be fitted to machine learning model.

Chapter 4

Approach

This chapter describes how we are going to approach the issues identified in the problem statement section of the introduction chapter. This chapter will also address how the rest of the work will be carried on.

4.1 Objectives

The main objective of this thesis is to investigate if there is any kind of relationship between internal operating temperature, power consumption and CPU load in the ALTO cloud. We also will be developing the simple monitoring system over IBM cloud. The upcoming subsections of this chapters will explain how we plan to achieve this objective.

4.2 Experiments

In research 5, we planned to conduct just a single experiment to see the relationship between temperature and power consumption under the high CPU load. Each NUMA node was stressed for 10 mins. All 8 cores of NUMA node was stressed. After stress, there was no activity for 10 minutes by putting the script to sleep. After 10 minutes of sleep, next NUMA node was stressed. The process was continued unless all 8 nodes (0-7) were stressed. But we ended up performing another experiment on Research5. The details and purpose of the second experiment are explained in result chapter No experiment will be performed in ALTO cloud. We will just run a script to collect data from ALTO without performing any experiment.

4.3 Data collection

A script will extract the values explained in design chapter. The script parses the value from an output of the terminal command. It will be using python's inbuilt subproccess module. The script will have several functions. Each Function will extract the specific value that we want. Such as function *Getcpuload* will extract all the CPU related data (total CPU load and a load of individual CPU core). The script will contain a function to extract memory, disk, temperature, and power related data. These values will be written into separate local .CSV files.These CSV files will be used for analysis in a later stage. In case of Research5 data were collected every 30 seconds. No data was collected when the script was sleep. For ALTO cloud the data were collected every minute for about 11 days. Co-supervisor of this thesis Harek, who is one of the administrators of ALTO will run the script in ALTO. The script will be tested in Research5 test server multiple times to correctly parse the correct values before running in ALTO. The only difference between the functioning of the script that will run on Research5 and ALTO is that CPUs in Research5 will be stressed.

4.4 Monitoring Platform

We are reusing the IoT infrastructure Khagendra developed during his thesis[3] as Monitoring platform and extend it to show the various system metrics. Detailed implementation is explained in the implementation chapter.

1. **Sending Data to IBM cloud** The python script will send the data periodically to IBM cloud. The MQTT protocol will be used to send the data. The device which is subscribed to the topic mentioned in the script will publish the data over IBM cloud.
2. **Visualization of data** The visualization of the data published by the device will be accomplished by using Watson IoT dashboard. We will create various cards that will visualize the data. Implementation of visualization of data is explained in implementation chapter.

4.5 Relationship between temperature, power consumption, and CPU load

The primary objective of this thesis is to see if there is any kind of relationship between temperature and power consumption. We will be calculating a correlation between them to see if there is any. We will be breaking down CPU load into CPU load per NUMA node and calculate the correlation between temperature and power consumption. We will also be seeing a correlation between temperature and power. This test will be performed for both Research5 test server and production server ALTO. We will be using R for this purpose. We will also be calculating the standard deviation to see how consistent the data we collected are.

4.6 Machine learning

We will develop simple supervised machine learning model. Features of the model will be 8 values of temperature sensors and Label will be the total power consumption which is the sum of all 4 power sensor readings. We will be fitting the model with various algorithms explained in background chapter. We will be comparing the performance of all machine learning algorithms. We will be comparing the mean square error and coefficient of determination between the actual output and the predicted output.

Chapter 5

Implementation

In this chapter we will explain how the plan highlighted in design and approach is realized into action

5.1 Data collection

The python script will be used to parse the values that we will be sending to IBM cloud and also store in a local file for further analysis, correlation test, and machine learning purpose. The data that is identified in section 3.1.2 of Design chapter is parsed from an output of specific Linux command. The script has 6 functions 2 for extracting CPU load and 1 each for extracting memory usage, disk usage, power, and temperature. Each function returns a dictionary containing a corresponding value. Code snippet of Function 'Getcputemp', which returns temperature is listed below as an example. This function returns the dictionary in the form {'temp':'value in degree Celsius'}

```
def Getcputemp(self):
        jj = []
        tempd = OrderedDict()
        cmd = 'sensors|grep temp1|cut -d \' \' -f9|tr -d \'°C\''
        xx = subprocess.check_output(cmd, shell=True).splitlines()

        for i in xx:
            jj.append(float(i))

        for i in range(0, len(jj)):
            tempd['temp' + str(i + 1)] = float(jj[i])
        return tempd
```

The data returned by the function will be written into the file and same data will be sent to IBM cloud. Since we are writing data every 30 seconds and 1 minutes for research5 and ALTO respectively, the value from the dictionary{'key':'value'} is stored in the temporary list and the content of this list is finally written into the CSV file. The following code snippet writes temperature data to file.

```
tempfile='temp_'+'_uti.csv'
templist = []
cputemp = Mytest()
dd = cputemp.Getcputemp()
  for i, j in dd.items():
    templist.append(j)

with open(tempfile, 'a') as d:
  writer = csv.writer(d)
  writer.writerow(templist)
  d.close()
```

5.2 Experiment

The Single experiment was conducted on research5 and no experiment was conducted in ALTO. Every NUMA nodes were stressed for 10 mins one after another with 10 minutes of gaps between every test. The gap between stress test was to provide enough time for CPU to cool and return to normal thermal state after a stress test.We binded numa node with *numactl* command. We binded all the numa node from 0 to 7. We stressed all 8 cpu cores of numa nodes.During the stress we collected data every 30 second and wrote it into the file.The experiment lasted little more than 2 and half hour.We got some weird result from data we collected from Research5 from this experiment.So we conducted another experiment in Research5.Detail and Result of this new experiment will be explained in 6.2.2 .

```
cmd=[]
for i in range(0,8):
  cmd.insert(i,'numactl --cpunodebind='+str(i)+' '+ 'stress -c 8 -t 600')


for i in cmd:
  timeafter10=datetime.datetime.now() + datetime.timedelta(minutes = 10)
  subprocess.Popen(i,shell=True)
  while(datetime.datetime.now<=timeafter10):
      #call functions and get the values
      #write those value into the file
      time.sleep(30)
  sleep(600)
```

We ran the python script for 11 days in ALTO as cron job.The script will executeevery minute, it will write the data to CSV file everytime it executes

5.3 Implementation of Monitoring platform over IBM cloud

The Next step in an implementation of the project was to set up the cloud platform so that the data extracted by the script can be sent for visualization.

Creating Cloud Foundry app

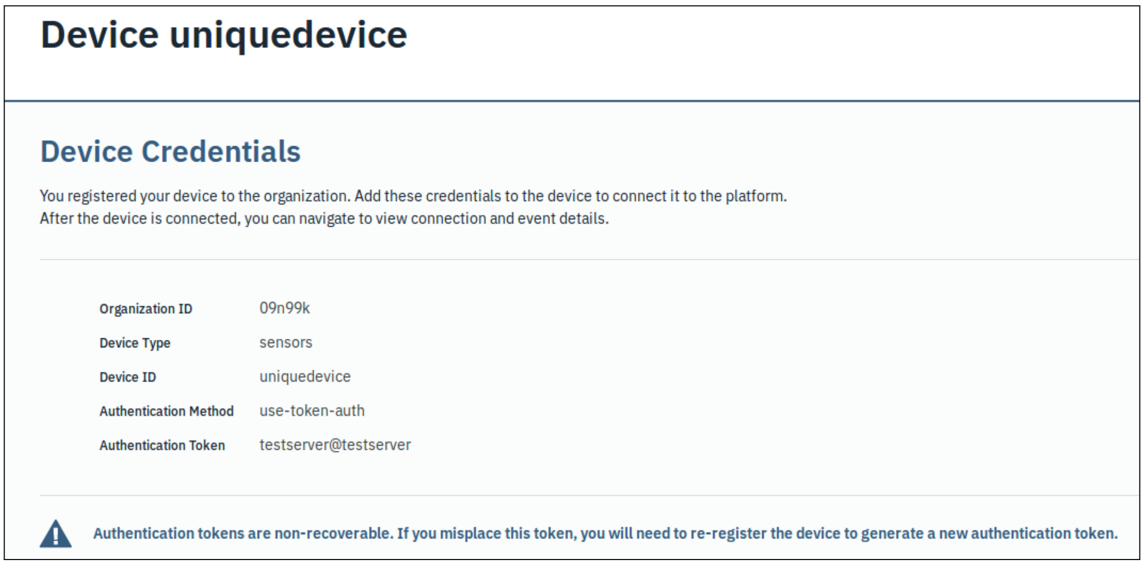
1. The first step was to sign up for the 6 months free account from IBM academy. After that username password combination can be used to log into IBM cloud at https://www.ibm.com/cloud/
2. Click on my cloud console -> dashboard on the top right panel. Screen prompting username and password will appear. Provide login credentials to log into IBM cloud console dashboard
3. Click on Create resource button. The resulting page will show various platforms and Infrastructure available. From platform select boilerplates and from boilerplates select internet of thing starter.
4. Now we need to create Cloud Foundry app. Provide unique App name. Free Lite plan does not allow the app to be deployed in multiple regions. The default region/location to deploy should be fine to get going. If the default location is not working try changing the location. Our location of deployment was the United Kingdom. The domain will reflect the area of deployment. Free plan will provide up to 256 MB of memory and apps are put to sleep after 10 days of development inactivity. This issues can be solved by upgrading to paid standard plan. Click on create button to create the cloud foundry app. In our case our app name is altomon for the alto server.

Creating a device that will publish data

- Once Cloud foundry app is created, getting started page will appear. Click on the overview tab. It will show a general overview of an app. Now under connection tab, there will be 2 options as *<appname>-cloudantNoSQLDB* and *<appname>-iotf-service*. Click on *<appname>-iotf-service* (**altomon-iotf-service** in our case)
- Click on the launch button to launch the Watson IoT platform. The Watson IoT dashboard will appear.
- Click on device -> add Device. Then click on device types -> add device types. Select the type (Device or Gateway). For the purpose of this thesis, we select device. Provide a device type name. We create a device type with name sensors. Click next. additional information can be provided about the device type but it

can also be left black. We left it blank in our thesis. Finally, click on a done button to create device type.

- Now its time to register the device to the device type we just created. Click on Register devices. Provide unique device id. This device id is local to cloud foundry app and have to be unique just within the app. Click next to continue. Device information may be supplied but as with device type we left it blank
- We are not associating the device with any groups so we just click next to continue. Now we need to provide authentication token. Auto-generated tokens can also be used. But we chose to create it. This token is encrypted by IBM before storing. Click next then done to create the device. The resulting page will show device credential as in the screenshot in a figure5.3. These credentials will be used with the script that will send data to this device.


Device uniquedevice

Device Credentials

You registered your device to the organization. Add these credentials to the device to connect it to the platform.
After the device is connected, you can navigate to view connection and event details.

Organization ID	09n99k
Device Type	sensors
Device ID	uniquedevice
Authentication Method	use-token-auth
Authentication Token	testserver@testserver

Authentication tokens are non-recoverable. If you misplace this token, you will need to re-register the device to generate a new authentication token.

Figure 5.1: Device credentials[1]

5.3.1 Sending data to IBM Cloud

We are using MQTT to send the data to IBM cloud. We have already created the device within IBM Waston IoT platform which publishes the data. The Python script sends the data to IBM cloud. The MQTT client connection is created and the data is published in IBM cloud. We are publishing 5 different sets of values (one each for temperature,

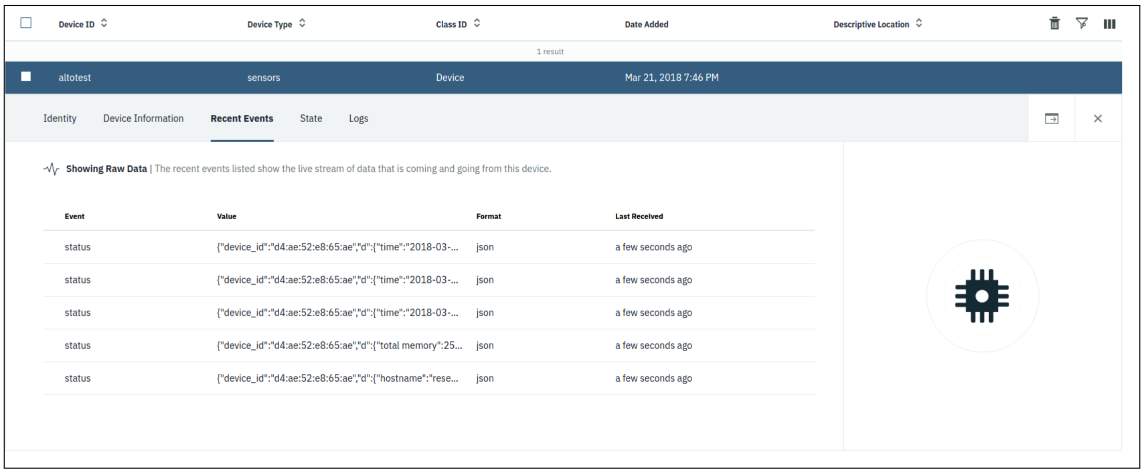
[1]This kind of Device was created number of times during the project. screenshot is just for illustation purpose.This device was not used to publish the data that we are using in this thesis.

power, CPU load, memory usage and disk usage). The following code listing from the final version of code ran in ALTO implements this task. This code was extracted from khagendra's thesis [3] and modified and extended to fit our purpose. We are using the device credentials from the previous section on creating the connection.

```
import paho.mqtt.client as mqtt
#Set the variables for connecting to the iot service
broker = ""
topic = "iot-2/evt/status/fmt/json"
username = "use-token-auth"
password = "altomonitor" #auth-token
organization = "qmznxv" #org_id
deviceType = "sensors"
deviceid ="altotest"

topic = "iot-2/evt/status/fmt/json"

#Creating the client connection
#Set clientID and broker
clientID = "d:" + organization + ":" + deviceType + ":" + deviceid
broker = organization + ".messaging.internetofthings.ibmcloud.com"
mqttc = mqtt.Client(clientID)

#Set authentication values, if connecting to registered service
if username is not "" :
    mqttc.username_pw_set(username, password=password)
    mqttc.connect(host=broker, port=1883, keepalive=60)


mqttc.loop_start()
#
#store the value to send in dictionary
#aa for temperature,bb for cpu
ac=':'.join(re.findall('..', '%012x' % uuid.getnode()))
##publishing to bluemix cloud
msgaa = json.JSONEncoder().encode({"d" :aa,"device_id" :mac})
mqttc.publish(topic, payload=msgaa, qos=1, retain=False)

msgbb = json.JSONEncoder().encode({"d": bb, "device_id": mac})
mqttc.publish(topic, payload=msgbb, qos=1, retain=False)
msgcc = json.JSONEncoder().encode({"d": cc, "device_id": mac})
mqttc.publish(topic, payload=msgcc, qos=1, retain=False)
msgdd = json.JSONEncoder().encode({"d": dd, "device_id": mac})
mqttc.publish(topic, payload=msgdd, qos=1, retain=False)
msgee = json.JSONEncoder().encode({"d": ee, "device_id": mac})
mqttc.publish(topic, payload=msgee, qos=1, retain=False)
```

Listing 5.1: Implentation of mqtt and sending data to IBM cloud

Once the script starts running it begins publishing data on IBM.The data is sent as JSON and can be seen in Recent events tab of the device.



Device ID	Device Type	Class ID	Date Added	Descriptive Location
altotest	sensors	Device	Mar 21, 2018 7:46 PM	

1 result

Showing Raw Data | The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
status	{"device_id":"d4:ae:52:e8:65:ae","d":{"time":"2018-03-...	json	a few seconds ago
status	{"device_id":"d4:ae:52:e8:65:ae","d":{"time":"2018-03-...	json	a few seconds ago
status	{"device_id":"d4:ae:52:e8:65:ae","d":{"time":"2018-03-...	json	a few seconds ago
status	{"device_id":"d4:ae:52:e8:65:ae","d":{"total memory":25...	json	a few seconds ago
status	{"device_id":"d4:ae:52:e8:65:ae","d":{"hostname":"rese...	json	a few seconds ago

Figure 5.2: Screenshot of data published in IBM cloud

5.3.2 Storing data in database

1. From the dashboard, click on the Cloud Foundry Services <appname>-cloudantNoSQLDB.

2. Click on launch to launch Cloudant Dashboard.Then click on a database on the left panel. Finally, click on Create Database and provide a name for the database. We named it *memdatabase*

3. Click on the cloud foundry app from the main dashboard and click on *visit app URL*.

4. Click Next and provide Username and password for Node-Red flow. Click Next and then Finish.

5. Now click on *Go To your NodeRed flow editor* to launch Node-Red. Provide Node red login credentials.

6. Drag the *ibmiot* from input *function* from function and *cloudant* from storage into the Node-Red flow editor.

7. connect these to each other. Wiring is illustrated in figure 5.3.2.

8. Double click on a function and write the function as

```
msg.payload = msg.payload.d ;
return msg ;
```

   Once connected the database starts storing the values published by the *device*

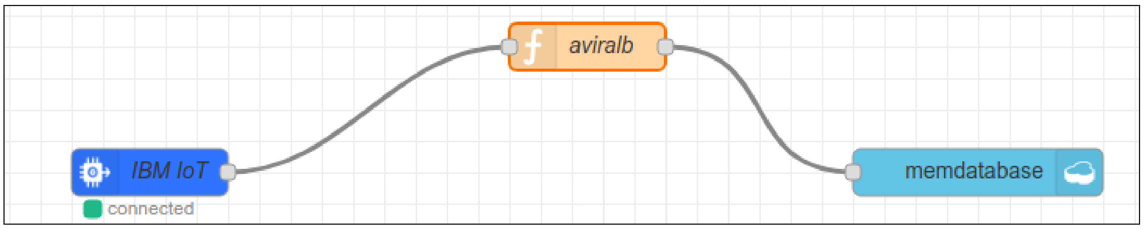


Figure 5.3: Node red wiring for database

5.3.3 Visualization of Data

As a part of thesis we are extending khagendra's work to build simple monitoring system.

- Log in IBM cloud console.bluemix.com and launch IBM Watson platform.
- Click on Board » *create new board* and provide name for the board. We named our board as ALTO board.
- Click on the newly created board and click again on the *Add new card* to add card to board. Number of cards can be added to board.Now choose the kind of visualization. IBM provides various visualization tools such as Bar Chart, line chart Donut chart etc. We added a card each to visualize power information, temperature information, CPU utilization information, Memory and Disk usage information. We used Donut chart for power, Bar chart for temperature and line chart for rest. Figure 5.3.3 and 5.3.3 are the screenshot of temperature and power consumption visualization of ALTO over IBM cloud.

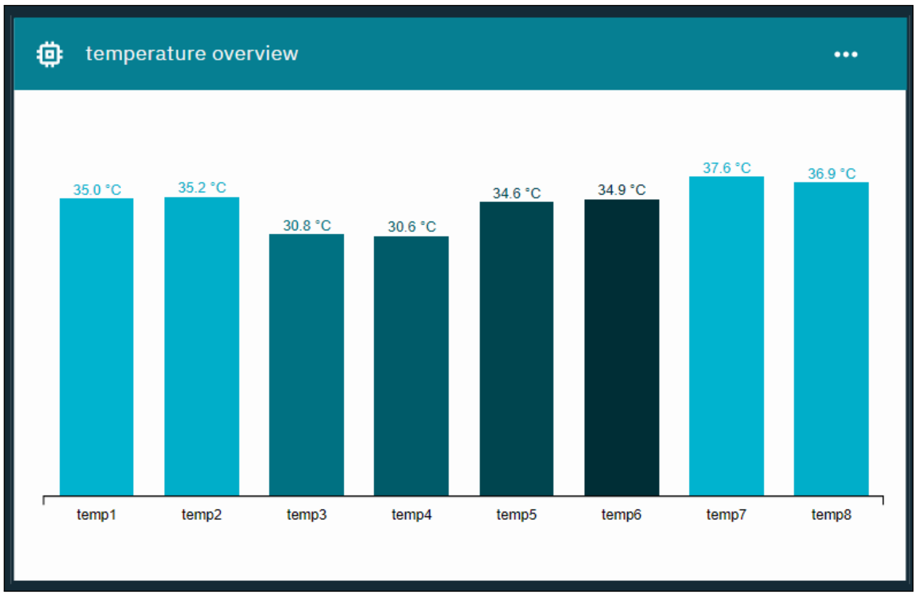


Figure 5.4: Visualization of temperature reading from ALTO over IBM cloud

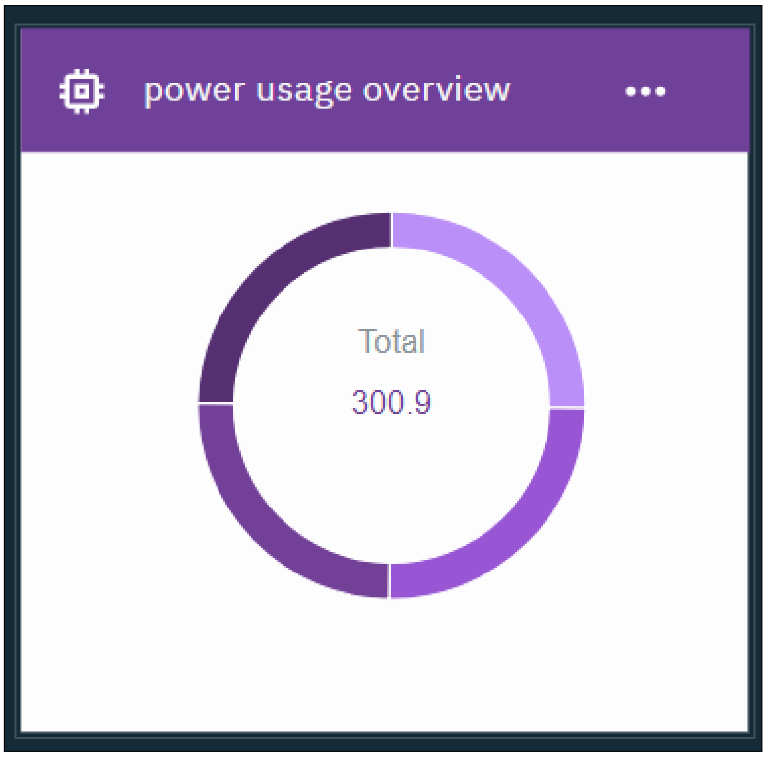


Figure 5.5: Power consumption of ALTO visualization over IBM cloud

5.4 Correlation Test

Rstudio, the IDE for R was used as a tool perform correlation test. The separate correlation test was performed for ALTO and research5. Finally, the correlation plot was generated to visualize the outcome of the test. Headers were added to CSV files that we collected from research5 and ALTO. There were 152 data for each variable for Research5 and 15044 for ALTO cloud. We first merged power data and temperature data into a single CSV file and the merged CSV file was used to perform correlation along with *cpu load* CSV. R's cor (x,y) function calculates the correlation between variable x and y.

First CSV file containing CPU load was loaded and then split into 8 different data frames.The resulting data frame contains the CPU load of the individual core of Numa nodes. Similarly, the merged temperature and power data were also split into 2 different data frames. Then, the correlation was calculated between these data frames. Also, a correlation test was performed against total power consumption with CPU loads and temperatures. Total power consumption was calculated by summing up values from all four variables of power data frame.The following R code snippet illustrates the splitting into sub data frame and calculation of correlation between temperature and power and CPU load of numa0 and temperature.Result of correlation test will be discussed in result and analysis chapter6.2

```
#loading a csv file containing CPU load
cpu=read.csv(file.choose('/home/aviral/cpufilealto.csv'),header=TRUE,sep=',')
#splitting the cpu data frame into 8 sub data frame per numa node
numa0=cpu[,c(4:11)]  # core 0-7
numa1=cpu[,c(12:19)] #core 8-15
numa2=cpu[,c(36:43)] #core 32-39
numa3=cpu[,c(44:51)] #core 40-47
numa4=cpu[,c(52:59)] #core 48-55
numa5=cpu[,c(60:67)] #core 56-63
numa6=cpu[,c(19:27)] #core 16-23
numa7=cpu[,c(28:35)] #core 24-31
temppow=read.csv(file.choose('/home/aviral/alto_pow_temp.csv'),header=TRUE,
    sep=',')
temp=temppow[,c(1:8)]#temperature
pow=temppow[,c(9:12)]#power
totalpower=temppow\$total#total power
temppowcor=cor(temp,pow)# correlation between temperature and power
corrplot(temppowercor, method='number'
numa0temp=cor(numa0,temp)# correlation between numa 0 cpu load and
    temperature
corrplot(numa0temp, method='number')
```

Listing 5.2: Correlation in R

5.5 Machine learning

The Machine learning is implemented using *scikit learn*. We are using 5 different machine learning algorithms, Linear Regression, Support vector machine, Decision Tree, Random forest and Artificial Neural Network. Our machine learning basically predicts total power consumption based on the internal temperatures reported by the output of *sensor* command. We are using the temperature and power data we collected from Research5 test server and ALTO cloud server. Our features are 8 temperature values and label is total power consumption which is calculated by summing up 4 power values we collected.We merged temperature and power data into a single CSV file.The new CSV file consists of 8 temperatures data, 4 power data and a data for total power consumption. Python's *pandas* data frame and *numpy* makes the task easier. We imported the CSV file as *pandas* data frame and created 2 *numpy* array that holds the label and features for our machine learning model. The code snippet from the final version of machine learning implementation illustrates how label and features were defined from pandas data frame.

```
import numpy as np
import pandas as pd
#loading the file into the panda dataframe
dfalto=pd.read_csv('temp_pow_alto.csv')
# drop the column that we dont need
dfalto.drop(['date'], 1, inplace=True)
dfalto.drop(['power1'], 1, inplace=True)
dfalto.drop(['power2'], 1, inplace=True)
dfalto.drop(['power3'], 1, inplace=True)
dfalto.drop(['power4'], 1, inplace=True)
#feature everything except total column
X_alto = np.array(dfalto.drop(['total'],1))
#label total column
y_alto = np.array(dfalto['total'])
```

Listing 5.3: Defining features and label for machine learning model

Next step was to split the data into training and testing data we split 80% of data as training data and remaining as testing data.*Scikit learn* provides easy way to split data into training and test sets.

```
from sklearn.model_selection import train_test_split
X_train_alto, X_test_alto, y_train_alto, y_test_alto =  train_test_split(
    X_alto,y_alto,test_size=0.2,random_state = 0)
```

The training and testing data set needs to be fitted into machine learning classifier. Except for ANN, other algorithms can be directly fitted into their corresponding classifier. We will have a separate subsection for implementation of ANN.

Once the data is fitted into machine learning classifier, we calculated the prediction on both test and training data set. The prediction was calculated for all of the machine learning algorithm's classifier used in this thesis. We calculated the mean square error between predicted values and the actual output (label). Mean square error is average of the differences between the estimator and what is being predicted[2]. Our estimator is the label value(Total power) in our data set. Finally, we are going to calculate the *r2_score* for the estimator and the predicted values. R2 or R_square is the measure of how close the data are fitted to the regression line.[3]. Higher the value, data fits better in the model. According to the *scikit learn* [14] documentation, negative *r2_score* indicates that model id worse for the fitted data. The listing 5.4 highlights the implementation of machine learning with Decision Tree

```
from sklearn.metrics import mean_squared_error,r2_score
from sklearn import tree

#defining machine learning classifier
dt = tree.DecisionTreeRegressor()
#fitting the data into the classifier
dt.fit(X_train_alto,y_train_alto)
#Make predictions on the training and test sets
DT_prediction_test = dt.predict(X_test_alto)
DT_prediction_train = dt.predict(X_train_alto)
#Report the errors in prediction on the train and the test set
DT_MRS_train_alto=mean_squared_error(DT_prediction_train,y_train_alto)
DT_MRS_test_alto=mean_squared_error(DT_prediction_test,y_test_alto)
DT_r2_score=r2_score(y_test_alto, DT_prediction_test))
```

Listing 5.4: implementation of machine learning using decision tree

5.5.1 Implementation of Artificial Neural Network

We are implementing ANN with *keras* with *tensorflow* as backend*Keras* is high-level neural network API for python. *Tensorflow* is also python library that allows expressing arbitrary computation as a graph of data flows [4]. Our first task was to build Artificial Neural Network architecture.

The listing 5.5 illustrates our Artificial Neural Network model. As input layer is going to be 8 temperature values and there are 4 hidden layers and an output layer. No of units in every layer is different in every layer. Figure 5.5.1 represents our model in much details. We are using sequential model for ANN. The sequential model is linear stack of layer. The input layer has 8 units corresponding to 8 temperatures. 4 different hidden layer has 10,5,10 and 1 units respectively.We are using relu(rectifier Linear unit) as activation function. The activation function introduces non liniearity to our model.According to wikipedia *"activation function of the node defines the output of that*

[2]https://en.wikipedia.org/wiki/Mean_squared_error accesed on may 10,2018

[3]http://blog.minitab.com/blog/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit

[4]https://www.oreilly.com/library/view/tensorflow-for-deep/9781491965320/ch01.html

node given input or sets of inputs.". Relu transforms the input in the range of 0 and input itself. f(x)=max(0,x)

We are compiling the ANN model adam optimizer and mean square error as loss function. Loss function measures difference between model's prediction and label. The Optimization algorithm is used to reduce the error.

```
import keras
from keras.models import Sequential
from keras.layers import Dense

# Make a neural network architecture
m,input_layer_size=X_alto.shape
NN_reg_alto = Sequential()
NN_reg_alto.add(Dense(input_dim = input_layer_size, units = 10, activation =
    'relu'))
NN_reg_alto.add(Dense(input_dim = input_layer_size, units = 5, activation = '
    relu'))
NN_reg_alto.add(Dense(units = 1))
NN_reg_alto.compile(loss='mean_squared_error', optimizer='adam')
```

Listing 5.5: Building ANN architecture

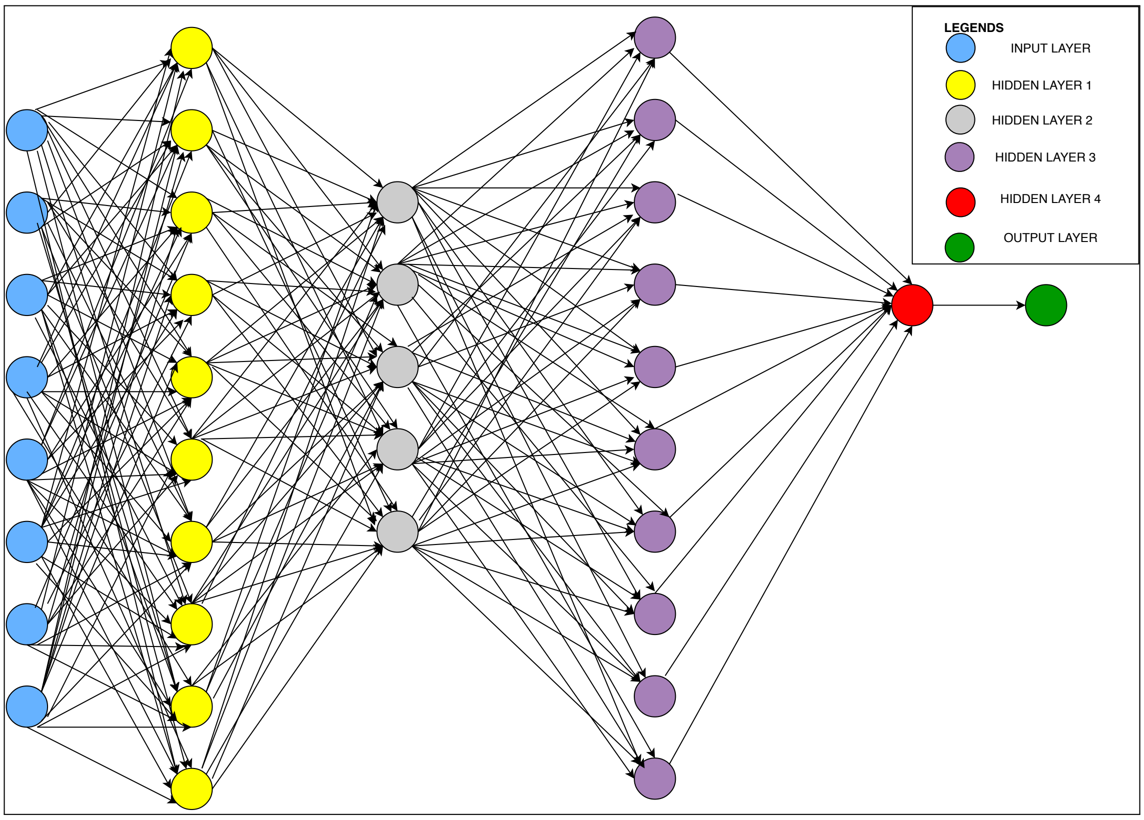


Figure 5.6: Artificial Neural Network Architecture Visualization

We are implementing the ANN model by same way we did for other algorithms.

```python
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.model_selection import train_test_split

NN_reg_alto.fit(X_train_alto, y_train_alto, batch_size = 150, epochs = 400,
    verbose=0)
#Make predictions on the training and test sets
NN_prediction_test_alto = NN_reg_alto.predict(X_test_alto)
NN_prediction_train_alto = NN_reg_alto.predict(X_train_alto)
#Report the errors in prediction on the train and the test set
NN_MRS_train_alto=mean_squared_error(NN_prediction_train_alto,y_train_alto)
NN_MRS_test_alto=mean_squared_error(NN_prediction_test_alto,y_test_alto)
```

Chapter 6

Result and Analysis

This chapter will explain and analysis the result from correlation test and machine learning in implementation section. CPU utilization and CPU load are used interchangeably in this section.Also CPUs and CPU cores or just cores are used interchangeably. We are using R5 to represent Research5 on the table 6.3 presented in this chapter

6.1 CPU utilization in ALTO

We were expecting at least some of the readings where CPU utilization to be around 40%, but we were proved wrong.Data we collected over period 11 days reveled there were only few data of over 10% of the total CPU utilization.For most part total CPU utilization was under 5%.Highest Total CPU utilization during test period was just under 16%.

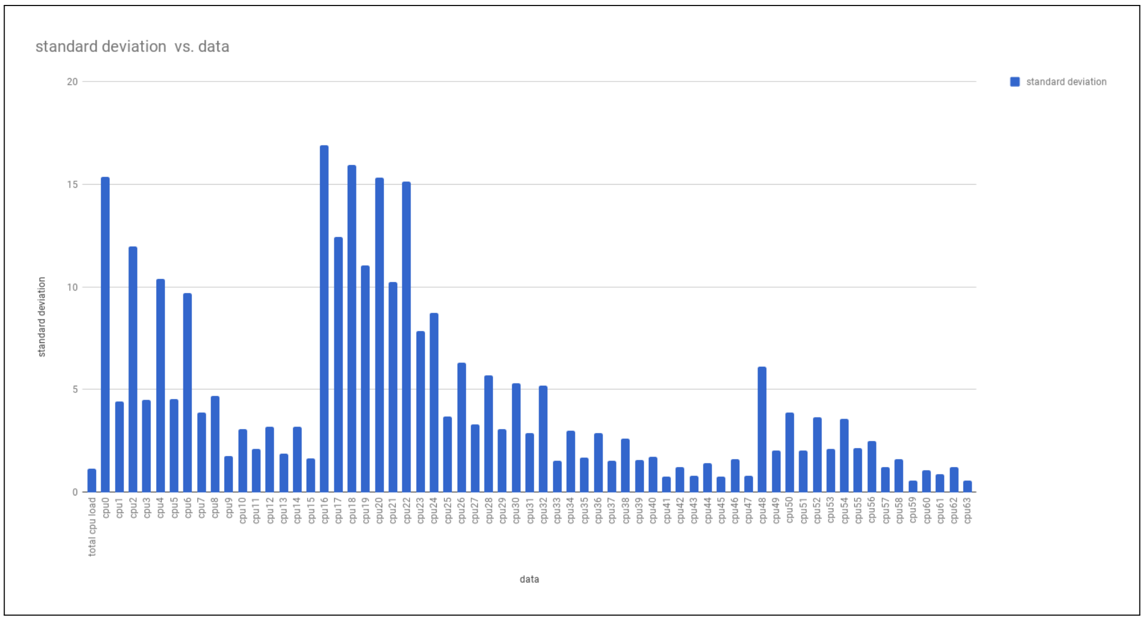


Figure 6.1: Standard deviation of CPU load in ALTO

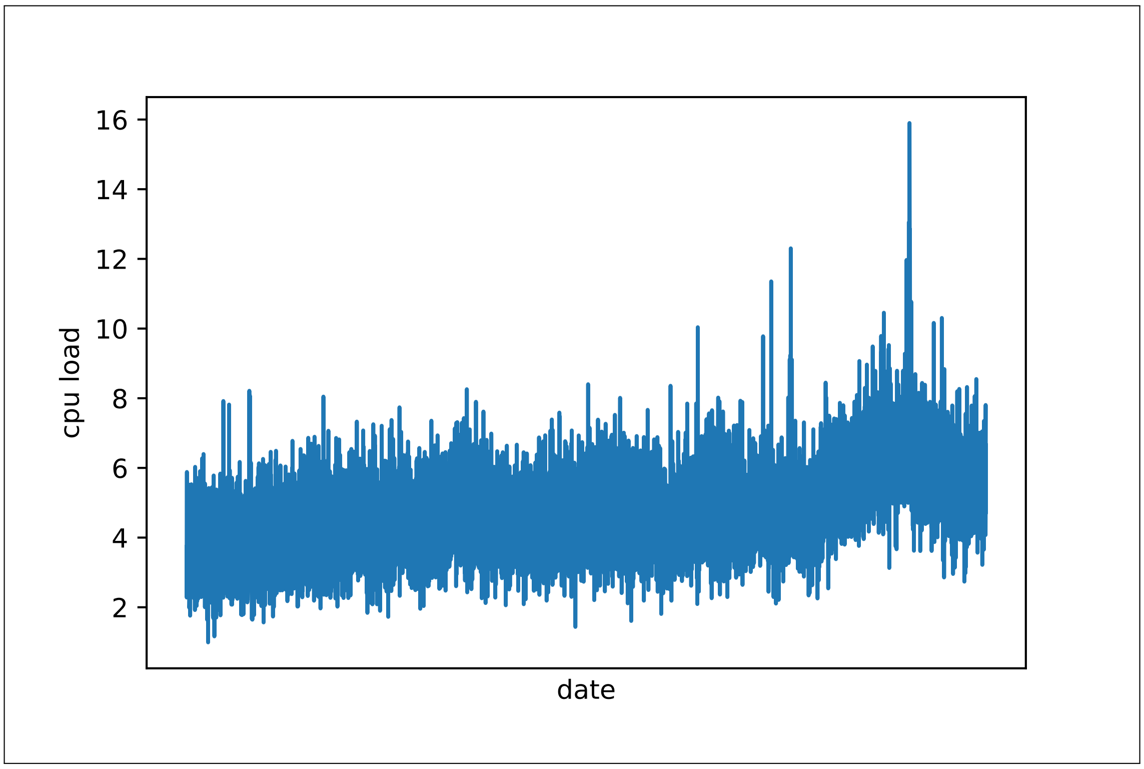


Figure 6.2: CPU utilization over 11 experiment days in ALTO

correlation value	Our classification
0 - 0.9	No correlation
0.1 - 0.19	Very weak
0.2 - 0.25	Weak
0.26 - 0.39	Observable
0.4 - 0.49	Moderate
0.5 -0.74	Strong
0.75+	Very strong

Table 6.1: Correlation result classification

We also noticed that some CPU cores were utilized more than other. Higher numbered CPU cores. We calculated the standard deviation6.1, the result showed some CPU cores have relatively higher standard deviation while other have significantly lower.

6.2 Correlation Test

This section presents the result of correlation test and some possible reasons behind the results.

6.2.1 Research5

All 8 CPU cores of all 8 nodes were stressed one after another in research5. So it only represents the extreme condition when there is (almost) 100 % CPU utilization. Another aspect to note is that Research5 is an experimental server with no external load.

6.2.1.1 Correlation between temperature and power

Four different temperature sensors were correlated to 4 different power sensors. Temperature sensors 1,3,5 and 7 are correlated to power sensors 1,2,3,and 4 respectively.The Correlation value is in between Temperature sensor 4,6 and 8 had a slightly weaker correlation with power sensor 2, 3 and 4 respectively. Remaining combination of temperature versus power has very weak or no correlation. Figure 6.3 shows correlation plot between temperature and power in Research5.

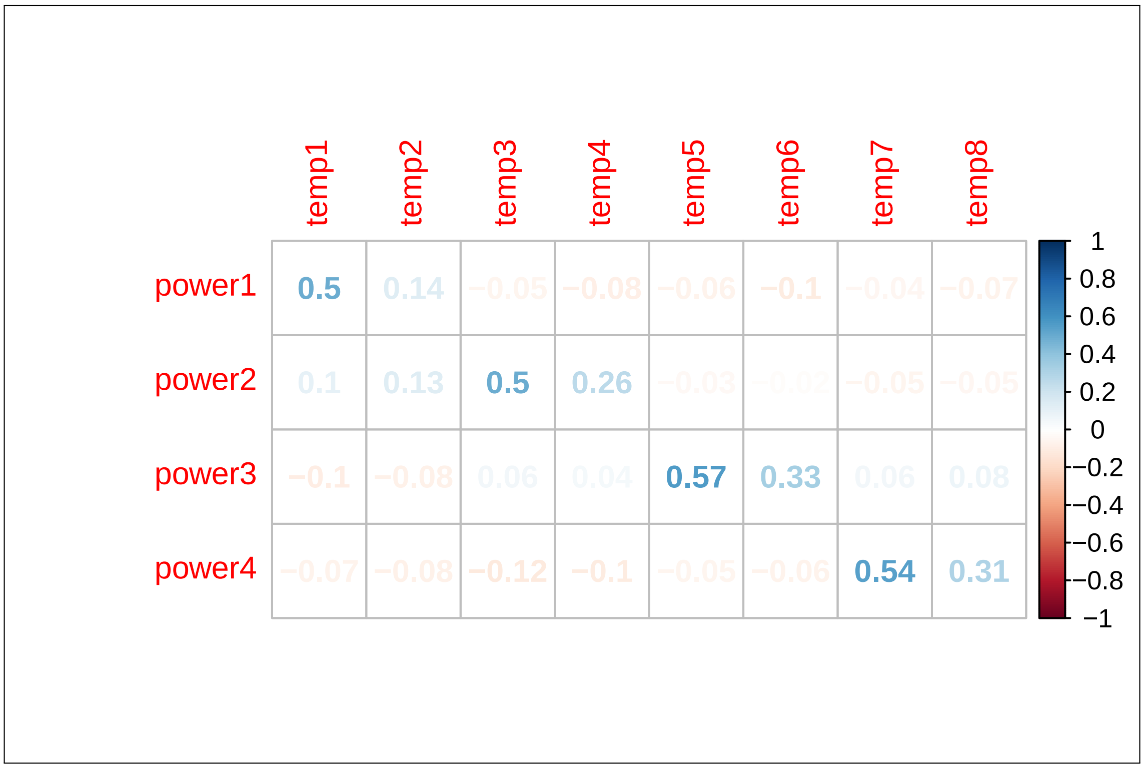


Figure 6.3: Correlation plot temperature vs power Research5

Analysis

Four different power sensors are affected highly affected by readings from 4 different temperature sensors. From the correlation plot, we can see lots of slight negative correlation and very weak correlation. We can assume that 4 different power sensors read power from 4 different parts of CPU. Research 5 has 4 CPU socket. It is likely that each of these power sensors is responsible for measuring CPU power consumption of different CPU socket. It might be like Power sensor1 measures the power consumption of Socket 1 and so on. Since there is one more temperature sensor which affects power sensor, we think it will be fair to assume that there is 2 temperature sensor per socket which is responsible for measuring the temperature of that particular socket. Correlation of 0 or near 0 supports this explanation. We mailed AMD asking about the CPU architecture but they did not share with us.

6.2.1.2 Correlation between CPU load and power

CPU load is split into CPU load per NUMA node. We got a strange result from correlation test between CPU load and power consumption. NUMA 0 has a very strong correlation with a Power1 but NUMA 1 has a strong negative correlation with Power1, NUMA 0 and NUMA 1 have no correlation with remaining of the power values. We see the same pattern of strong positive correlation and strong negative correlation with one of the power values but no correlation with others. CPU cores in even-numbered NUMA node have strong positive correlation while odd have a strong negative correlation. NUMA node 2 and 3 have a correlation with power 2, NUMA node 4 and 5 with power 3 and NUMA 6 and NUMA 7 with Power 4. Pattern continued when correlation with total power was calculated. Only difference was the correlation values fall on *moderate* category as opposed to *strong* and *very strong*.

Analysis

The reason behind strange result we are getting might be with the inconsistency of our data collection.We did not collect the data when the NUMA node was let to cool.

6.2.1.3 Correlation between CPU load and temperature

There were mixed result when Correlation between CPU load and temperatures was calculated for research5.Since all the Cores within the NUMA node were stressed. CPU load Correlation value against temperature is almost same for all the cores.

- NUMA 1 has no correlation with Temperature 1, moderate with temperature and weak negative with the remainder of temperatures.
- NUMA 2 has no correlation with temperature 1 and Temperature 2, very strong correlation with temperature 3, strong with temperature 4 and weak negative with remaining of the temperatures.
- NUMA 3 has a weak correlation with temperature 3, moderate correlation with temperature 4 and very weak negative correlation with rest of the temperatures.
- NUMA 4 has a very strong correlation with temperature 5, strong with temperature 6, weak very negative correlation with temperature 1, temperature 2, temperature 3 and temperature 4. NUMA 4 has no correlation with temperature 7 and temperature 8.
- NUMA 5 has a weak correlation with temperature 5, moderate correlation with temperature 6, no correlation with temperature 8 and very weak negative correlation with a remainder of the temperatures.

- NUMA 6 has a very strong correlation with temperature 7, strong with temperature 8 and very weak negative with remaining of the temperature.
- NUMA 7 has a very weak correlation with temperature 7 and moderate with temperature 8. It has a very weak negative correlation with remaining of the temperature.

Analysis

We have mixed result.There is a strong or very strong correlation with some of NUMA nodes and temperatures. When there is correlation, there is correlation with just 1 or 2 temperatures. We can not draw any conclusion from this result as we have already seen some inconsistency with collected data from the previous sections.

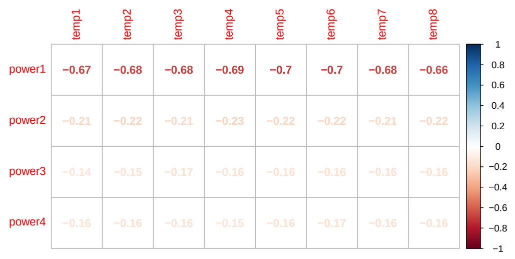
6.2.2 Second experiment in Research5

After we got some inconsistent and weird results from the experiment in Research5, we decided to perform a new experiment. We are going to gradually increase the CPU load all the way to 100 percent. Instead of stressing all the CPUs within NUMA node, this time we created load on 1 CPU at a time. We conducted the experiment varying CPU load. We simulated 0% CPU load and we increased the load by 25% all the way to 100%. This means that we stressed with 0%, 25%, 50%, 75% and 100% CPU utilization. Every CPU is stressed 5 mins each for varying CPU load (5 mins of 0 percent, 5 mins for 25 percent, 5 mins for 50 percent, 5 mins for 75 percent and 5 mins for 100%). There will 2 minutes of a gap when moving from one CPU load to other. There will also be 5 minutes of a gap when switching from one CPU to another. We are using the python script CPULoadGenerator by Gaetano Carlucci [1]. The script generates a finite amount of load on particular CPU for a certain amount of time. The script makes the Use of PI controller to achieve the task of generating fixed CPU load on particular CPUs for a specified amount of time. We are going to collect data every 30 seconds like we did in first experiment but this time we are also going to collect data during the period when CPU is left unstressed to cool down.

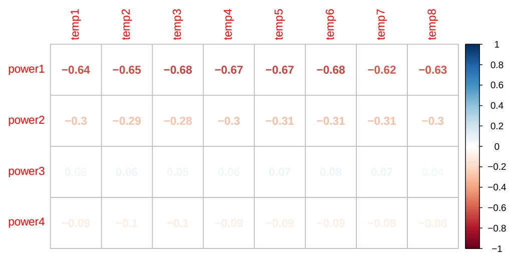
6.2.2.1 Correlation between temperature and power

The result was completely different from the first experiment. Unlike strong correlation that we perceived in the first experiment, we got no correlation. Even weirder we got correlation values mostly in the very weak negative zone. We calculated correlation between temperature Because of time limitation, we randomly selected 8 pairs (16

[1]https://github.com/GaetanoCarlucci/CPULoadGenerator



(a) CPU11



(b) CPU12

Figure 6.4: Correlation Plot between power and temperature for CPU11 and CPU12 Research5 experiment 2

CPUs) and calculated the correlation. For most of CPUs we tested, there was a negative correlation often in the range of moderate to strong 6.2.2.1.

6.2.2.2 Correlation between CPU Load and temperature

From the data we collected from second experiment in Research5, there was no correlation between CPU Load and temperature.

6.2.2.3 Correlation between CPU load and power consumption

There was either no correlation or very weak negative correlation between CPU load and power consumption.

6.2.2.4 Analysis of Experiment 2 in Research5

We performed the second experiment in Research5 because of we found strong negative correlations between temperature and power for some NUMA nodes. The results from the second experiment were not as expected. It revealed no correlation between any combination of CPU load, power, and temperature. It is a weird observation. Also, we noticed a negative correlation between temperature and power when it was calculated in relation to specific CPU. We also noticed lots of correlation values on a very weak negative zone. Our theory is that temperatures are staying in the relatively low range because more power is consumed to keep the system cool. For eg. temperature might be around 36-degree Celcius, after sometimes it goes down to say 35 degrees even through CPU load is almost same, this happens because of CPU fans running at higher speed to consume the heat accumulated in the heat sink. So it might be the case that power consumption is similar or higher for the relatively low temperature. This kind of behavior may have lead to a scenario where there is no or some sort of negative correlation. Although individual CPU was tested with CPU utilization up to 100 percent, overall CPU utilization was below 4 percent during the entire duration of the second experiment in Research5. It is not unusual to have no relationship between temperature and power under low CPU utilization.

6.2.3 ALTO

We collecting real-life data from ALTO.We are not artificially creating the load. We did not get the result as per our assumption as highest CPU load recorded over a period of 11 days and from 15044 data points was just under 16%. It was way below our expectation.

6.2.3.1 Correlation between temperature and power

Only power3 has the noticeable correlation with all 8 temperature. Power2 has a very weak correlation with temperature. Similarly, power1 has a weak correlation with temperature1 and an even weaker correlation with temperature2.It has no correlation with remaining of the temperature sensors data. Power4 has a relatively higher correlation with temperature 7 and temperature 8, No correlation with temperature1, very weak with temperature2 and correlation of around 0.2 with rest of the temperatures.The correlation plot in figure 6.5 displays results of this correlation test. When the correlation between total power consumption and temperatures was analyzed, there was some correlation. 6.6

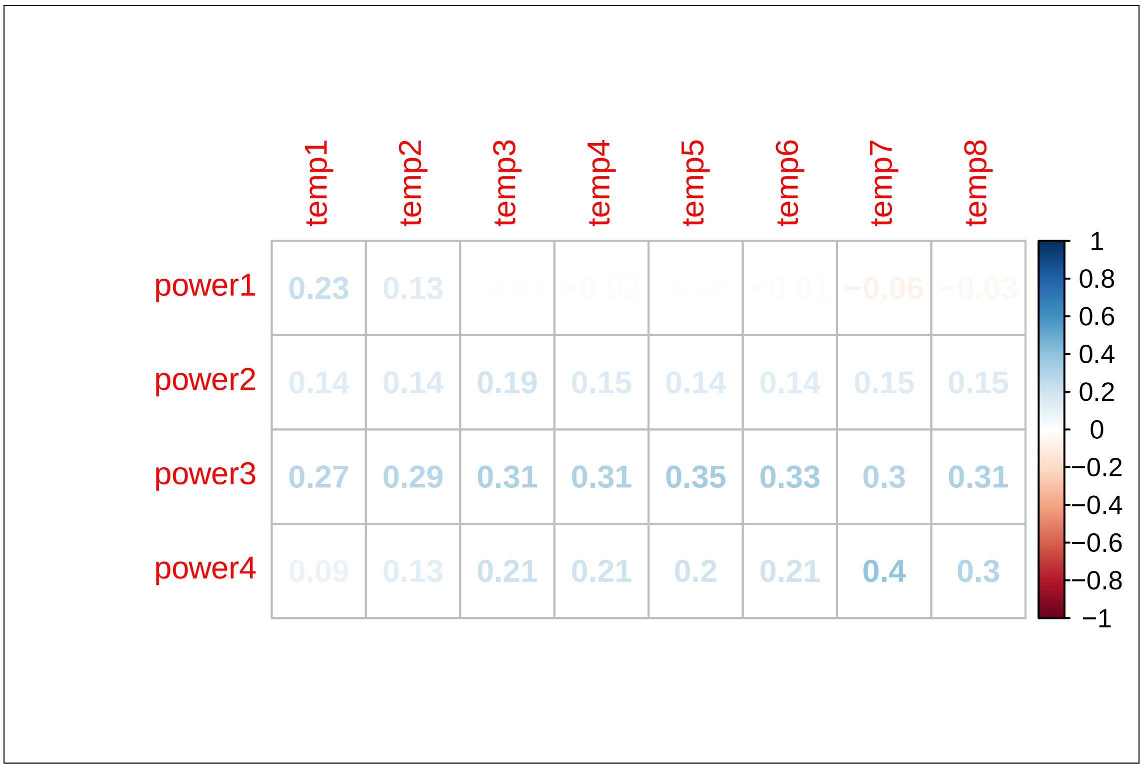


Figure 6.5: Correlation plot power vs temperature ALTO

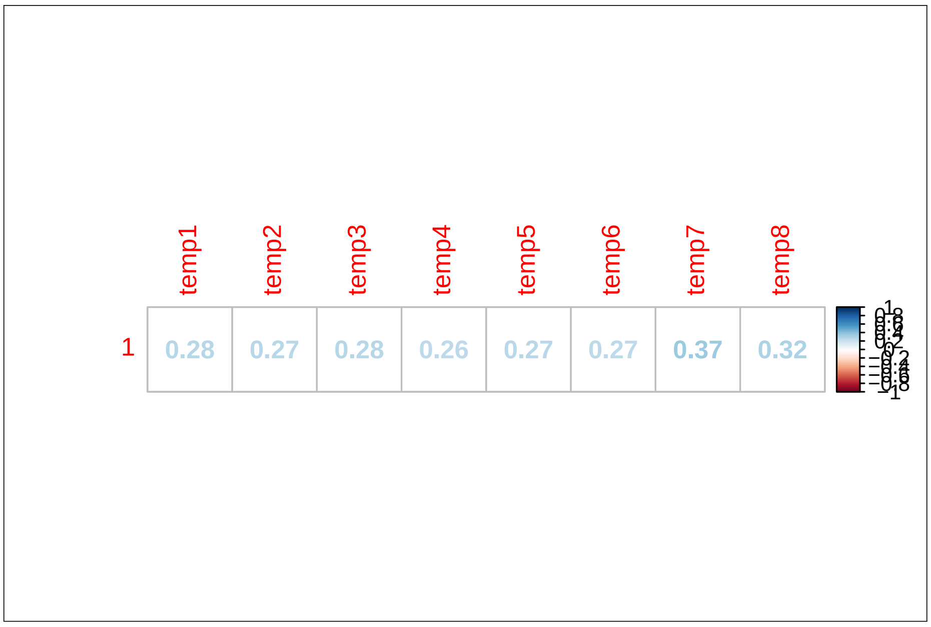


Figure 6.6: Correlation plot Total power consumption vs Temperature

Analysis

Correlation plot in 6.5 show very little correlation between power and temperature (apart from power3 vs temperature).

- Apart from weak correlation with temperature1 and temperature2,Power1 does not have a correlation with temperatures. It is extremely unlikely that temperature will have some sort of visible impact on readings from power sensor 1.
- Power2 has very weak positive correlation with temperature. Increasing temperature is very less likely to increase readings of power sensor 2 and vice-versa
- Power 3 has an observable correlation with all the temperature values. Rise on readings of any of the temperature sensor will result in a rise of reading on power sensor 3 and vice-versa
- Power4 has mixed results from our correlation test. Temperature7 and Temperature 8 have a higher impact on the output of Power sensor 4.
- Although the temperature might not have that much effect on the individual power reading . But it has impact on total power consumption.Increase in temperature means higher power consumption. Although according to result of our correlation test, ratio may be less significant, meaning that increasing temperature does not increase total power consumption by the same amount.
- Since We only got the highest CPU utilization of around 16 % from the data we collected every minute for about 11 days, it will be fair to say that this kind of behaviour may not be representative of case when server is under high load.

6.2.3.2 Correlation between CPU load and power

There is no correlation between CPU load and power. There is a correlation within observable range between Power3 and CPU load in NUMA 4. Rest has no correlation.
Analysis
Since we don't have the CPU load that went above 16 percent during 11 days of our experiment period. We believe the load was not sufficient enough to show noticeable change in power reading

6.2.4 Correlation between CPU load and temperature

The correlation test between the CPU load and the temperature ended up with mixed results.

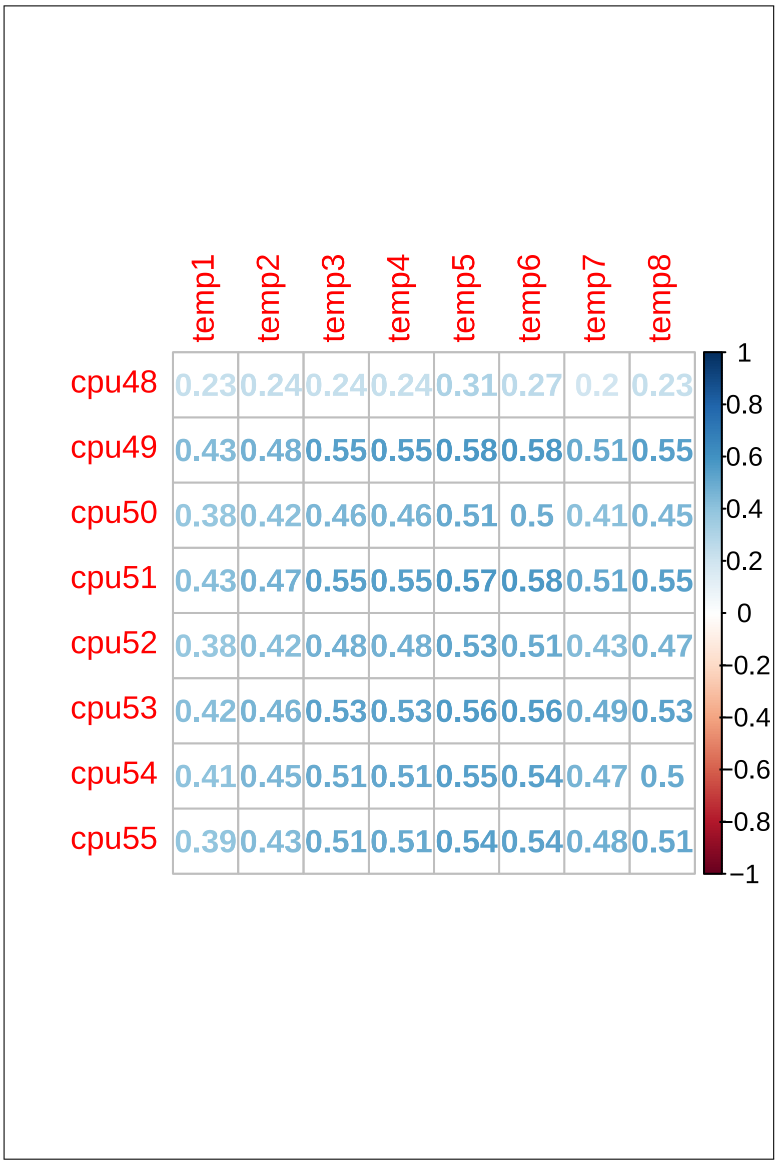


Figure 6.7: Correlation plot CPU utilization of NUMA 4 vs Temperatures, it shows strong correlation although NUMA 4 has low average CPU utilization

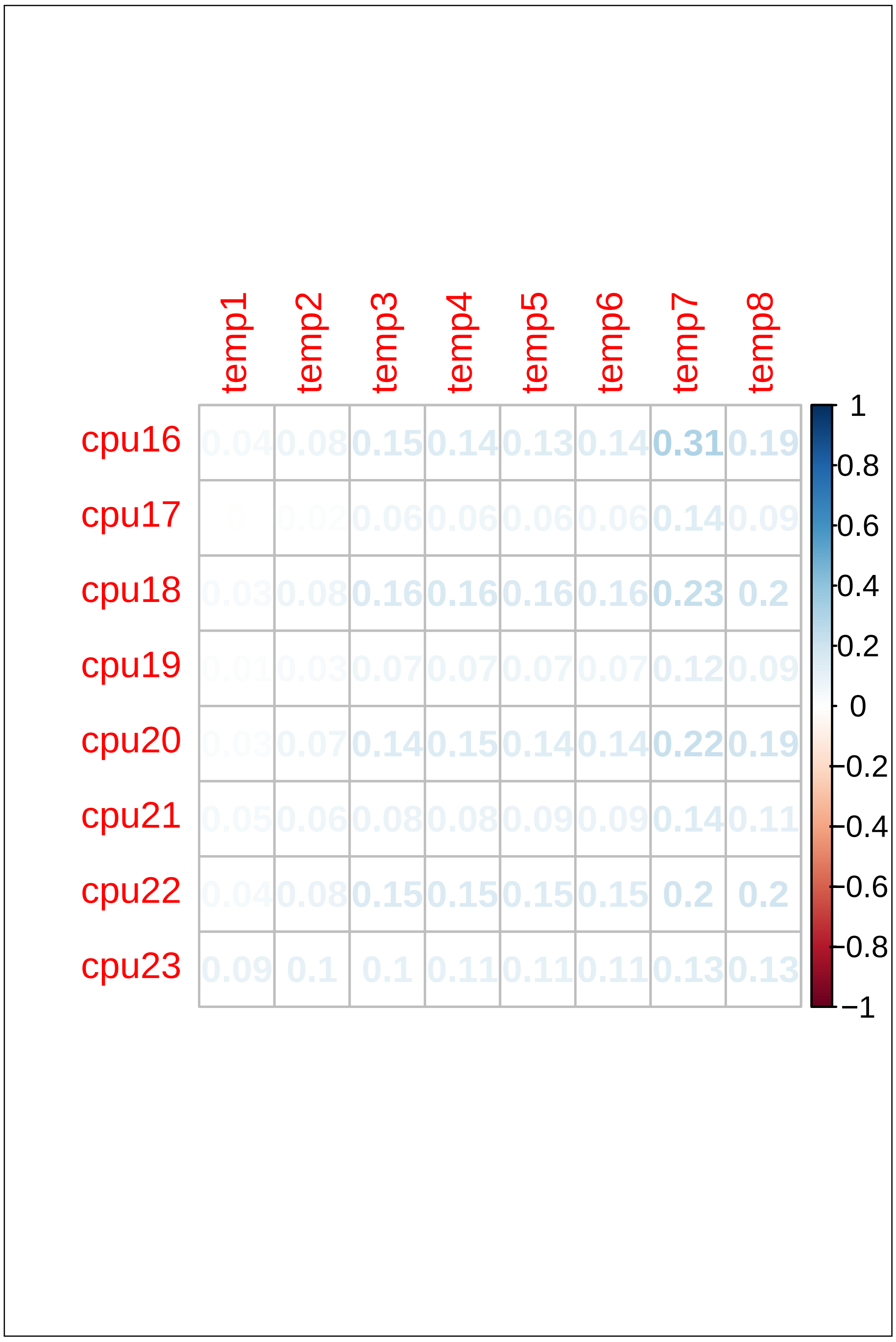


Figure 6.8: Correlation plot NUMA 6 vs Temperatures, it shows weaker correlation although NUMA 6 has high average CPU utilization than other nodes

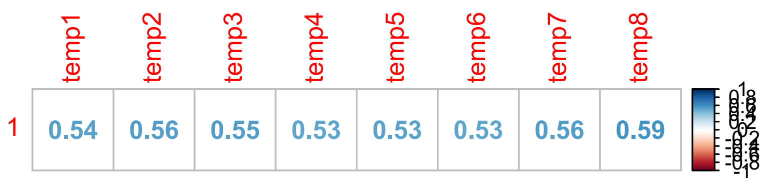


Figure 6.9: Correlation between CPU utilization and temperatures

- CPU loads in NUMA 0 and NUMA 3 have no correlation with temperature. All CPUs have a correlation equal or less than 0.1 with temperatures in NUMA 3. Temperature1 and Temperature 2 have few very weak correlations with CPU load in NUMA 0 and others have no correlations.
- CPU load in CPU 10 and CPU 12 of NUMA node 1 have a correlation of 0.25 with temperature 2. These CPUs have a weak correlation with rest of the temperatures. CPU 8 has a weak correlation with Temperature 1 and Temperature 2 and very weak correlation with rest of the temperatures. On the other hand CPU 9, 11 and 15 have a very weak correlation with temperatures. CPU 13 has no correlation except the very weak correlation with temperature 1.
- Temperature sensors 3 have observable correlation with all the CPUs of NUMA 2. CPU 32 of NUMA node 2 has very weak correlation with remaining of the temperature values we collected from ALTO. The correlation values between CPU cores of NUMA 2 and temperatures are uniform apart from CPU 32 are uniform. 7 temperature values (except temperature 3) have correlation with CPU load that are mostly around weak zone 6.2.
- NUMA 5 has the very weak correlation.NUMA 7 has a weak correlation. CPU 16 of observable correlation with temperature 7. CPU 18,20 and 22 have a weak correlation with all 8 temperatures.
- Unlike other NUMA nodes, NUMA node 4 has moderate to a strong correlation.Only CPU 48 weaker correlation in this NUMA node. Most of the CPU load vs temperature correlation range between 0.45-0.55 which indicates relatively strong correlation.

- Total CPU Utilization has a strong correlation with temperature [6.8]. It has the correlation of over 0.5 with all 8 temperatures.This reflects that although the loads on the individuals core may not have noticeable effect on CPU temperature,but with,the higher CPU utilization,more heat will be generated.

Analysis

NUMA 4 had a strong correlation with temperature while other had weaker and some had almost no correlation. We calculated the average CPU load of all the 64 CPU cores and compared with the cores within the NUMA 4 to see if this NUMA node had a higher workload than other nodes. We found that average load in all the CPUs were less than average CPU utilization. CPUs with higher average CPU utilization have a far low correlation with temperature. The CPUs in NUMA 6 and NUMA 1 has higher average CPU load than other nodes. NUMA 3 had no correlation with temperatures. The average CPU load of all the CPUs in NUMA 3 were less than 1 %. We believe that the loads in on CPUs of NUMA3 were not enough to have any kind of effect on temperatures. It was little surprise to see NUMA 4 has the higher correlation with temperatures given that this NUMA node was used very less compared to some other. NUMA 4 have a strong correlation with temperature although over 95 % of the CPU load we collected for the CPU cores (core 48-55) were less than 4%. This indicates that lighter load in these cores can also result in an increase in temperature. One possible explanation might be Socket 3 is not close to cooling fans. NUMA 4 is in Socket 3. NUMA node 0,1 and 7 have higher CPU load but they contributed less in generated heat. These CPUs might be close to cooling fans.

6.3 Machine learning

We are creating the machine learning model to see if we can predict total power consumption just by looking at the internal temperature. We fitted the data with 5 different supervised machine learning algorithms, Linear regression, Support vector machine, Decision Tree, Random forest and Artificial Neural Network. We were not expecting our model to be highly efficient one and the results proved it. We model fit better with the data from Research5 but did not fit well for ALTO. Figure 6.10 and 6.11 presents the comparison of Mean square error for training and testing data set between various machine learning classifier.

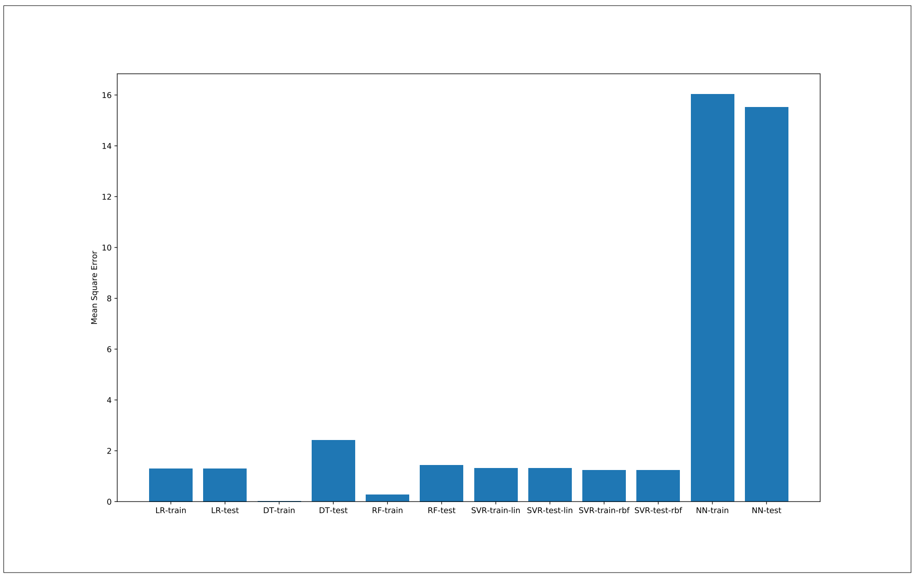


Figure 6.10: Comparison of mean square error for training and testing data set with different machine learning algorithm for ALTO

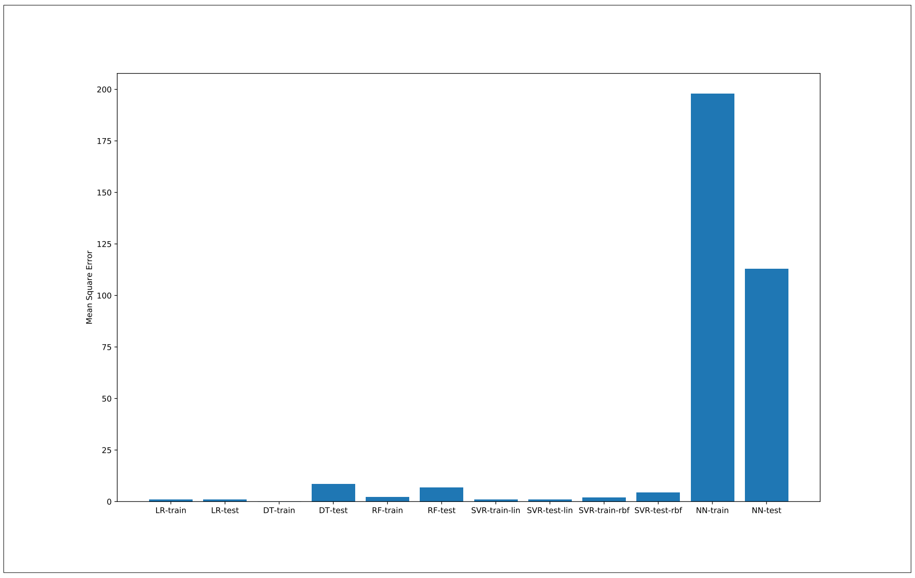


Figure 6.11: Comparison of mean square error for training and testing data set with different machine learning algorithm for Research5

Analysis

It is evident that ANN performed worse with our datasets. There is a huge difference in the mean square error between ANN and other algorithms. Mean square error of ANN for both test and training data is enormously huge for Research5 in excess of 100. Mean square error for both training and test datasets are less than 2.5 for remaining of the algorithms in case of research5. The error is close to 0 for training dataset for Decision Tree but surprisingly for the testing datasets, the error is over 8. Linear regression and SVM with linear kernel have an error of around 1.5 for both test and training dataset. Comparision of error for the various algorithm on Research5 is illustrated in figure 6.11. For research5 SVM and Linear Regression have smallest of the error between estimator and predicted value.This holds true for both of the experiments.

ANN again has biggest error for ALTO but this time its significantly less compared to that in Research5.In ALTO, Mean square error for training dataset is again near to 0 for Decision Tree and around 0.5 for Rainforest Algorithm but testing dataset's error for these 2 algorithms is significantly higher in comparison to training datasets. There is a clear pattern of rule-based decision-making classifiers have a very low error for training datasets but higher error when it comes to testing datasets. All other algorithms have an error of just above 1.5 for both training and test datasets. Mean square error

We calculated the *r2 score* to see how well our data fit to the different machine learning classifiers. Best possible *r2 score* is 1. More the value is closer to 1 better the data fits into the model and vice-versa. We noticed negative *r2 score* value for ANN. This means our data fitted worse for ANN. Decision Tree also have negative *r2 score* for ALTO testing data sets. Other classifiers have positive values but it is very low positive values. For Research5 result is positive and is more than 0.5 except for ANN which has negative *r2 score*. Some classifiers have the value close to 1. This indicates that besides ANN, our data fitted quite well with other classifiers. Table 6.3 illustrates the *r2 score* of various algorithms in both ALTO and Research5. We will analyze this behaviour in next subsection.

Algorithms/Classifier	ALTO	R5 1st experiment	R5 2nd experiment
ANN train	-8.65	-4.9	-34362.63
ANN test	-8.36	-2.9	-35252.38
Linear Regression train	0.2	0.96	0.54
Linear Regression test	0.18	0.96	0.45
SVM RBF train	0.23	0.92	0.54
SVM RBF test	0.21	0.86	0.57
SVM linear train	0.18	0.91	0.44
SVM linear test	0.17	0.89	0.54
Decision Tree train	0.99	0.99	0.98
Decision Tree test	-0.48	0.64	0.1
Random Forest train	0.84	0.94	0.88
Random Forest test	0.11	0.64	0.45

Table 6.2: R2_score for of different ML algorithms

6.3.1 Why data fitted nicely for Research5 but not for ALTO

The table 6.3 shows that apart from ANN and decision tree testing in the second experiment in Research5, data fitted quite nicely for Research5 but not for ALTO. Figure 6.12 shows a comparative overview of how data fitted in the second experiment on Research5. We believe strong fitting of data in Research5 has something to do with the way experiment was approached and the way data was collected. Experiment in Research5 was conducted by stressing the CPUs with 100 percent CPU load for a certain amount of time. The data were collected periodically during the stress period. We did not collect any data when CPUs were left to cool down. Since we are collecting the data when part of the processor (all cores within a NUMA node at a time) was stressed, the data we collect may have been of a similar pattern. This is evident by the strong correlation between temperature and Power in Research5 (Although there was an only correlation between particular temperature and power value eg. Temp1 with power1, it was the strong one). The data fitted nicely in machine learning model (Yes, except for ANN) because they were collected under similar condition (CPU load) and ML algorithms were able to extract some kind of relationship between temperature and power data combinations. Although we changed our experiment to collect data when CPU was stressed with a different value, all the CPUs were stressed in a similar manner. Each of the 64 CPUs was stressed from 0% of CPU load and the load was increased by 25 percent until it reaches 100 percent. However, the scenario was completely different in ALTO. Unlike the controlled data collection as in Research5, we were collecting real-world data. There was a huge difference in operating scenario. Although we got maximum CPU utilization of just around 16 percent, ALTO operated under higher CPU load differential with low being as low as around just 1 percent. Data we collected from ALTO represents the data under varying CPU load, whereas in Research5 it was under same CPU load of 100 percent. When we developed the machine learning model, we were not expecting it to be a very good model. Given that there was some relationship between internal operating temperature and power consumption, Our model should predict the total power consumption from the input temperature data. We knew that this prediction was never going to be perfect.

How successfully the machine learning model performs depends on actually how good the model is. The good model takes account of all the parameters that may influence the final outcomes (Total power consumption in our case.) Power consumption not only depends on CPU load but other several factors like Disk and Memory Usage, Surrounding ambient temperature, cooling system, and many others. In case of ALTO, it shares the room with other servers too. So, the operation of other servers and their heat output will also affect ALTO. Our goal was to develop a simple model that takes the parameters that are readily available to the system and study the relationship between temperature and power consumption. Purpose of our machine learning model was to investigate how precise we can predict the power consumption just analyzing the internal temperatures. Another reason, our model might not have performed as expected is that we did not get data we expected. With low CPU load, the heat produced might have been absorbed by heat sink and temperature may not have increased by a noticeable amount. This may have lead to the scenario where for similar temperature values with slightest of the difference there was a high differential in power values.

6.4 Logical vs Physical cores, Which one is better?

Neither data from ALTO nor Research5 revealed anything that we can use to answer this question. Based on the data we collected there was no evidence that running the process on logical CPU consumes more power or physical CPU or vice-versa. There is no distinguishable difference in temperature when running a job in logical and Physical CPUs.

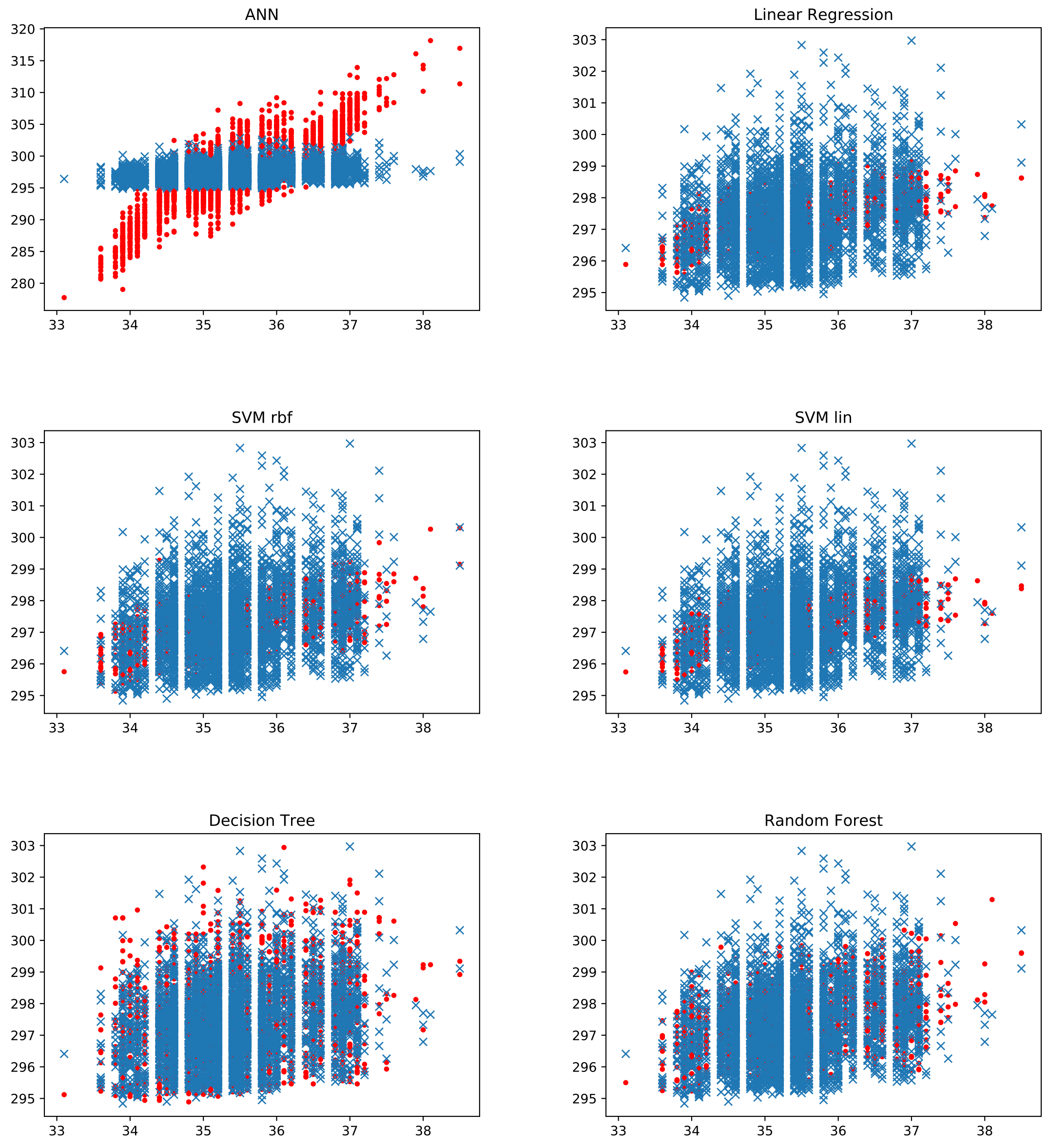


Figure 6.12: How our model fitted to different machine learning algorithm for ALTO

Chapter 7

Discussion

The aim of our study was to investigate relationship between temperature, power and cpu load. We also reused Khagendra's work [3] and extended it to visualize the system resource utilization data that we collected during the project.This thesis was not just limited to finding the relationship between temperature and power consumption but we were also interested to discover if running a job in a logical processor consume less or more power compared to physical one. We opted different approach to carry out our experiments. We based our work on the live data collected from the production server ALTO instead of running the simulation and drawing the conclusion based on the output of the simulation.

However results we got were not as expected.*Ideal* results would have been the cases where we could have drawn some fitting relationship between temperature and power consumption. That's why there is the word *ideal*, it rarely happens. Still, we were optimistic to discover some conclusive relationship between temperature and power. The general understanding is that temperatures in computer system are the function of resource usage given that operating environment remains constant. There should be increase in temperatures with an increase in resource utilization. High usage of resources such as a processor, a disk and other requires more power.Therefore, considering the general transitive relation, -temperature and power consumption should be related to each other. The server room is air-conditioned and maintained almost at a constant temperature. Our research domain is narrowed to study of internal temperature and power consumption in relation to CPU load. In theory, our research has very less or no relation with external power consumption. The correlation study suggested there might be some weak correlation between temperature and power. However, this does not provide enough evidence to reach any conclusion. According to our study,a strong correlations between temperatures and CPU load was observed but not between CPU load and power consumption.The correlation result we observed in Research5 only represents the result for a specific case. Although we observed the strong correlation between CPU load and power in first experiment, the second experiment in Research5 which was more precise showed no correlation between CPU load and power consumption. Further, The results from machine learning algorithms as well were not as expected. Before running machine learning tests we expected Artificial Neural Network to provide us with higher preciseness.In contrary, the results -we got were the exactly opposite to what we expected. The data did not fit for ANN . We got a negative r2 score for all 3 data sets (2 from Research5 and 1 from ALTO) on which machine learning model was fitted. We expected some differences in errors and accuracy of a model between various machine learning algorithms. Nevertheless, the results we obtained were rather surprising as there was a huge difference between ANN and other algorithms.

Looking back on the project, certain aspects regarding the approach could have been improved. Results from machine learning proved that our model was not the very good model. Some other parameters such as temperatures within different parts of server rack and ambient temperatures ,if taken into account might have improved the results . However, our goal was to find some relationship between internal temperatures and power consumption under the realistic operating scenario. Such findings could then be used to make constructive suggestions to save electricity costs. Another considerations for our study could have been disk and memory utilization. Yet, disk and memory data we collected showed that their utilization was on the lower side. Although we did not deviate from our original research goal, inclusion of the above-mentioned parameters in our research would have broaden the research domain and brought improvements to our results. One of the reasons why we did not get the expected result was because of our data. We did not get the data as per our expectation at first place. Our data was mostly with very low CPU utilization. The highest CPU load we collected over 11 days was just 15.86 percent. The result we are getting from correlation tests and machine learning may not be bad at all. Our data is a representative of data under low resource utilization. Therefore, there might be no such visible relation between CPU utilization with power consumption or temperature with power consumption that can be mapped.

Part III

Conclusion

Chapter 8

Conclusion and Future Work

Data revealed that there is a strong correlation between CPU load and temperature in a production server ALTO. The CPU utilizations were on the lower side for ALTO. So result we achieve may or may not hold true for high CPU utilization or very low utilization. Experiment in test server Research5 revealed the mixed result. There was some correlation between temperature and power consumption under high CPU load but there was no correlation between Power and temperature in relatively very low total CPU utilization. Our study suggested that impact of the individual CPU utilization is not observable in temperature and power consumption. Surprisingly there is no correlation between CPU load and total power consumption based on over 15000 data points collected over the period of 11 days.

Based on the result we achieve, thermal-aware workload placement mechanism might reduce the internal temperature but there is no guarantee that it will reduce power bills. Power consumption in the cloud-based server is not only dependent on CPU utilization and internal temperature. The result of the machine learning suggested that if some workload placement was to be developed, not only internal temperatures but other factors also should be considered. There is no such relation between internal temperature and power consumption within the server that can be precisely mapped.

Our conclusion is based on the data that did not have a single value of high CPU utilization. So it might not be same when a processor has higher utilization. As a suggestion of future work, we would strongly recommend the way to observe the temperature and power characteristics under relatively higher CPU utilization. In addition to CPU utilization, the relationship between other resources utilization such as GPU can be studied with power consumption and temperature. We also recommend to include ambient temperature and temperature of the various area of server racks such as top, bottom, and side, if somebody might be interested to build machine learning model to establish a relationship between power consumption and temperatures.

Bibliography

[1] Azure. *Azure defination cloud*. Feb. 2018. URL: https://azure.microsoft.com/en-in/overview/what-is-cloud-computing/.

[2] Lakshmi N. Bairavasundaram et al. 'An Analysis of Latent Sector Errors in Disk Drives'. In: *SIGMETRICS Perform. Eval. Rev.* 35.1 (June 2007), pp. 289–300. ISSN: 0163-5999. DOI: 10.1145/1269899.1254917. URL: http://doi.acm.org/10.1145/1269899.1254917.

[3] Khagendra Basnet. 'IoT Based Temperature Sensors Monitoring for Smart Workload Distribution on Cloud '. MA thesis. Oslo, Norway: University of Oslo, 2017.

[4] Tom Bawden. *Global warming: Data centres to consume three times as much energy in next decade, experts warn*. Feb. 2018. URL: https://www.independent.co.uk/environment/global-warming-data-centres-to-consume-three-times-as-much-energy-in-next-decade-experts-warn-a6830086.html.

[5] Stephen J. Bigelow. *Hypervisor*. Mar. 2018. URL: https://searchservervirtualization.techtarget.com/definition/hypervisor.

[6] D. J. Bradley, R. E. Harper and S. W. Hunter. 'Workload-based power management for parallel computer systems'. In: *IBM Journal of Research and Development* 47.5.6 (2003), pp. 703–718. DOI: 10.1147/rd.475.0703.

[7] Michael Bullock. *Data center definition*. Feb. 2018. URL: https://www.cio.com/article/2425545/data-center/data-center-definition-and-solutions.html.

[8] Cisco. *Layered architecture data center*. Feb. 2018. URL: https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Data_Center/DC_Infra2_5/DCInfra_1.html.

[9] copperking. *Please explain svm like I am 5 years old*. May 2018. URL: https://www.reddit.com/r/MachineLearning/comments/15zrpp/please_explain_support_vector_machines_svm_like_i.

[10] Thomas H. Davenport. *What are some popular machine learning methods?* Mar. 2018. URL: https://www.sas.com/en_us/insights/analytics/machine-learning.html.

[11] M. Dayarathna, Y. Wen and R. Fan. 'Data Center Energy Consumption Modeling: A Survey'. In: *IEEE Communications Surveys Tutorials* 18.1 (Firstquarter 2016), pp. 732–794. ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2481183.

[12] IBM docs. *About IBM watson*. Mar. 2018. URL: https://console.bluemix.net/docs/services/IoT/iotplatform_overview.html#about_iotplatform.

[13] Pivotal documentation. *components of cloud foundry*. Mar. 2018. URL: https://docs.run.pivotal.io/concepts/architecture/.

[14] Scikit Documentation. *sklearn.metrics.r2$_s$core*. May 2018. URL: http://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html/.

[15] Niklas Donges. *The random forest algorithm*. https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd. Last accessed 21 April 2018.

[16] Luke Dormehl. *What is an artificial neural network*. Mar. 2018. URL: https://www.digitaltrends.com/cool-tech/what-is-an-artificial-neural-network/.

[17] Virginia Fernandez. *How does bluemix works*. Mar. 2018. URL: https://www.slideshare.net/VirginiaFernandez11/how-does-ibm-bluemix-work.

[18] Gartner. *Cloud defination*. Feb. 2018. URL: https://www.gartner.com/newsroom/id/1035013.

[19] Gartner. *Gartner definition data center*. Feb. 2018. URL: https://www.gartner.com/it-glossary/data-center/.

[20] Mark golden. *Data centers can slash CO2 emissions 88% or more*. Feb. 2018. URL: https://energy.stanford.edu/news/data-centers-can-slash-co2-emissions-88-or-more.

[21] Migjen Hakaj. *The InteliRack project*. Mar. 2018. URL: https://www.duo.uio.no/bitstream/handle/10852/51846/Master_thesis_Migjen.pdf?sequence=1&isAllowed=y.

[22] Ben Hatton. *Data center tiers*. Feb. 2018. URL: https://www.thedatacave.com/data-center-tiers-explained.

[23] IBM. *IBM defination cloud*. Feb. 2018. URL: https://www.ibm.com/cloud/learn/what-is-cloud-computing.

[24] William Koehrsen. *Random forest simple explanation*. https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d. Last accessed 21 April 2018.

[25] Mathworks. *Machine Learning*. Mar. 2018. URL: https://www.mathworks.com/discovery/machine-learning.html.

[26] mqtt.org. *What is MQTT*. Mar. 2018. URL: http://mqtt.org/faq.

[27] Bhumi Nakhuva and Tushar Champaneria. 'Study of Various Internet of Things Platforms'. In: 6 (Dec. 2015), pp. 61–74.

[28] Andrew Ng. *What is machine learning*. Mar. 2018. URL: https://www.coursera.org/learn/machine-learning/lecture/Ujm7v/what-is-machine-learning.

[29] Katherine Noyes. *facebooks-next-data-center-to-rely-only-on-renewable-energy*. Feb. 2018. URL: https://www.computerworld.com/article/2945045/data-center/facebooks-next-data-center-to-rely-only-on-renewable-energy.html.

[30] Opencv. *Introduction to svm*. Mar. 2018. URL: https://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html.

[31] Timothy Grance Peter Mell. *Cloud computing definations*. Feb. 2018. URL: http://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf (visited on 12/02/2018).

[32] Eduardo Pinheiro, Wolf-Dietrich Weber and Luiz André Barroso. 'Failure Trends in a Large Disk Drive Population'. In: *Proceedings of the 5th USENIX Conference on File and Storage Technologies*. FAST '07. San Jose, CA: USENIX Association, 2007, pp. 2–2. URL: http://dl.acm.org/citation.cfm?id=1267903.1267905.

[33] Saimadhu Polamuri. *HOW THE RANDOM FOREST ALGORITHM WORKS IN MACHINE LEARNING*. https://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/. Last accessed 21 April 2018.

[34] Jim Gao Richard Evans. *DeepMind AI Reduces Google Data Centre Cooling Bill by 40%*. Feb. 2018. URL: https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/.

[35] Margaret Rouse. *definition of cloud foundry*. Mar. 2018. URL: http://searchcloudcomputing.techtarget.com/definition/Cloud-Foundry.

[36] Margaret Rouse. *What is machine learning*. Mar. 2018. URL: https://searchenterpriseai.techtarget.com/definition/machine-learning-ML.

[37] Rahul Saxena. *How Decision Tree Algorithm works*. Mar. 2018. URL: http://dataaspirant.com/2017/01/30/how-decision-tree-algorithm-works/.

[38] Nosayba El-Sayed et al. 'Temperature Management in Data Centers: Why Some (Might) Like It Hot'. In: *SIGMETRICS Perform. Eval. Rev.* 40.1 (June 2012), pp. 163–174. ISSN: 0163-5999. DOI: 10.1145/2318857.2254778. URL: http://doi.acm.org/10.1145/2318857.2254778.

[39] Raffaele Stifan. *IBM Bluemix The Cloud Platform for Creating and Delivering Applications*. Mar. 2018. URL: http://www.redbooks.ibm.com/redpapers/pdfs/redp5242.pdf.

[40] Yevgeniy Sverdlik. *Apple datacenter 100% renewables*. Feb. 2018. URL: http://www.datacenterdynamics.com/content-tracks/design-build/apple-reaches-100-renewable-energy-across-all-data-centers/74708.fullarticle.

[41] Yevgeniy Sverdlik. *Data center energy consumption*. May 2018. URL: http://www.datacenterknowledge.com/archives/2016/06/27/heres-how-much-energy-all-us-data-centers-consume.

[42] Kenneth G. Brill W. Pitt Turner IV John H. (Hank) Seader. *Data center tiers*. Feb. 2018. URL: https://www.connectixcablingsystems.com/themes/ccs/pdf/Tier_Classification.pdf.

[43] University of Washington. *Machine Learning: Classification*. Mar. 2018. URL: https://www.coursera.org/learn/ml-classification.

[44] Stephan Watts. *SaaS vsPaaS vs IaaS*. May 2018. URL: https://www.bmc.com/blogs/saas-vs-paas-vs-iaas-whats-the-difference-and-how-to-choose/.

[45] Wikipedia. *Support Vector machine*. https://en.wikipedia.org/wiki/Support_vector_machine. Last accessed 21 April 2018.

Chapter 9

Appendix 1

9.1 Machine Learning Implementation

```
#for alto

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.model_selection import train_test_split
#Linear regression
from sklearn import linear_model
#svm
from sklearn.svm import SVR
#decision tree
from sklearn import tree
#rainforest
from sklearn.ensemble import RandomForestRegressor
#neural nets
from sklearn.ensemble import RandomForestRegressor
import keras
from keras.models import Sequential
from keras.layers import Dense
dfalto=pd.read_csv('temp_pow_alto.csv')
# drop the column that we dont need
# drop the column that we dont need
dfalto.drop(['date'], 1, inplace=True)
dfalto.drop(['power1'], 1, inplace=True)
dfalto.drop(['power2'], 1, inplace=True)
dfalto.drop(['power3'], 1, inplace=True)
dfalto.drop(['power4'], 1, inplace=True)
#feature everything except total column
X_alto = np.array(dfalto.drop(['total'],1))
#label total column
```

```
y_alto = np.array(dfalto['total'])
#spliting the data into training and testing set, with 80 percent data as
    training and 20 percent as testing
#ALTO
X_train_alto, X_test_alto, y_train_alto, y_test_alto =  train_test_split(
    X_alto,y_alto,test_size=0.2,random_state = 0)
#################
#####linear regression#############
#defining model
lr = linear_model.LinearRegression()
#fitting model
lr.fit(X_train_alto,y_train_alto)

#Make predictions on the training and test sets
LR_prediction_train_alto=lr.predict(X_train_alto)
LR_prediction_test_alto=lr.predict(X_test_alto)

#Report the errors in prediction on the train and the test set
LR_MRS_train_alto=mean_squared_error(LR_prediction_train_alto,y_train_alto)
LR_MRS_test_alto=mean_squared_error(LR_prediction_test_alto,y_test_alto)
##########################
######SVM################
#fitting the data
svr_rbf=SVR(kernel='rbf',C=2,gamma=0.1)
svr_lin=SVR(kernel='linear',C=2,gamma=0.1)
svr_rbf.fit(X_train_alto,y_train_alto)
svr_lin.fit(X_train_alto,y_train_alto)
#prediction
svr_rbf_train_predict=svr_rbf.predict(X_train_alto)
svr_rbf_test_predict=svr_rbf.predict(X_test_alto)

svr_lin_train_predict=svr_lin.predict(X_train_alto)
svr_lin_test_predict=svr_lin.predict(X_test_alto)

# calculating the error
lin_train_MSE=mean_squared_error(svr_lin_train_predict,y_train_alto)
lin_test_MSE=mean_squared_error(svr_lin_test_predict,y_test_alto)

rbf_train_MSE=mean_squared_error(svr_rbf_train_predict,y_train_alto)
rbf_test_MSE=mean_squared_error(svr_rbf_test_predict,y_test_alto)

########Decision Tree################
#defining model
dt = tree.DecisionTreeRegressor()
dt.fit(X_train_alto,y_train_alto)
#Make predictions on the training and test sets
DT_prediction_test = dt.predict(X_test_alto)
DT_prediction_train = dt.predict(X_train_alto)
#Report the errors in prediction on the train and the test set
DT_MRS_train_alto=mean_squared_error(DT_prediction_train,y_train_alto)
DT_MRS_test_alto=mean_squared_error(DT_prediction_test,y_test_alto)

###########RAINFOREST################
```

```python
rf= RandomForestRegressor()
rf.fit(X_train_alto,y_train_alto)
#Make predictions on the training and test sets
RF_prediction_test_alto = rf.predict(X_test_alto)
RF_prediction_train_alto = rf.predict(X_train_alto)
#Report the errors in prediction on the train and the test set
RF_MRS_train_alto=mean_squared_error(RF_prediction_train_alto,y_train_alto)
RF_MRS_test_alto=mean_squared_error(RF_prediction_test_alto,y_test_alto)
###########NEURAL NETS##########################
# Make a neural network architecture
m,input_layer_size=X_alto.shape
NN_reg_alto = Sequential()
NN_reg_alto.add(Dense(input_dim = input_layer_size, units = 10, activation =
    'relu'))
NN_reg_alto.add(Dense(input_dim = input_layer_size, units = 5, activation = '
    relu'))
NN_reg_alto.add(Dense(units = 10, activation = 'relu'))
NN_reg_alto.add(Dense(units = 1))
NN_reg_alto.compile(loss='mean_squared_error', optimizer='adam')
#spliting the data into training and testing set, with 80 percent data as
    training and 20 percent as testing
X_train_alto, X_test_alto, y_train_alto, y_test_alto =  train_test_split(
    X_alto,y_alto,test_size=0.2,random_state = 0)
NN_reg_alto.fit(X_train_alto, y_train_alto, batch_size = 150, epochs = 400,
    verbose=0)
#Make predictions on the training and test sets
NN_prediction_test_alto = NN_reg_alto.predict(X_test_alto)
NN_prediction_train_alto = NN_reg_alto.predict(X_train_alto)
#Report the errors in prediction on the train and the test set
NN_MRS_train_alto=mean_squared_error(NN_prediction_train_alto,y_train_alto)
NN_MRS_test_alto=mean_squared_error(NN_prediction_test_alto,y_test_alto)
#Make comparision
fig=plt.figure(figsize=(16,10))
myplt = [LR_MRS_train_alto, LR_MRS_test_alto,DT_MRS_train_alto,
    DT_MRS_test_alto, RF_MRS_train_alto, RF_MRS_test_alto,
        lin_train_MSE  ,lin_test_MSE,rbf_train_MSE,rbf_test_MSE,
    NN_MRS_train_alto, NN_MRS_test_alto]
plt.bar(range(len(myplt)), myplt)
plt.xticks(range(len(myplt)), ('LR-train', 'LR-test','DT-train','DT-test', '
    RF-train' , 'RF-test', 'SVR-train-lin','SVR-test-lin','SVR-train-rbf','
    SVR-test-rbf','NN-train','NN-test'))
plt.ylabel('Mean Square Error')
plt.savefig('comparision.pdf')
```

Similar approach was implemented for Research5 data.

9.2 Script to send data to IBM cloud and write to the file

```python
# -*- coding: utf-8 -*-
# !/usr/bin/python
from collections import OrderedDict
from uuid import getnode as get_mac
import collections
import time, subprocess
import os, time, signal
import threading
import paho.mqtt.client as mqtt
import json
import uuid
import re
from time import sleep
import datetime
import csv


class Test(object):
    def Getcpuinfo(self):
        '''
                http://stackoverflow.com/questions/23367857/
                accurate-calculation-of-cpu-usage-given-in-percentage-in-
    linux
                read in cpu information from file
                The meanings of the columns are as follows, from left to
    right:
                    0cpuid: number of cpu
                    1user: normal processes executing in user mode
                    2nice: niced processes executing in user mode
                    3system: processes executing in kernel mode
                    4idle: twiddling thumbs
                    5iowait: waiting for I/O to complete
                    6irq: servicing interrupts
                    7softirq: servicing softirqs

                #the formulas from htop
                     user    nice   system  idle      iowait irq   softirq
    steal  guest  guest_nice
                cpu  74608   2520   24433   1117073   6176   4054  0        0
          0      0


                Idle=idle+iowait
                NonIdle=user+nice+system+irq+softirq+steal
                Total=Idle+NonIdle # first line of file for all cpus

                CPU_Percentage=((Total-PrevTotal)-(Idle-PrevIdle))/(Total-
    PrevTotal)
                '''
        cpustat = '/proc/stat'
        sep = ' '
        sleeptime = sleep(1)

        cpu_infos = OrderedDict()  # collect here the information
        with open(cpustat, 'r') as f_stat:
            lines = [line.split(sep) for content in f_stat.readlines() for
line in content.split('\n') if
                    line.startswith('cpu')]

        for cpu_line in lines:
            if '' in cpu_line: cpu_line.remove('')  # remove empty elements
            cpu_line = [cpu_line[0]] + [float(i) for i in cpu_line[1:]]  #
type casting
            cpu_id, user, nice, system, idle, iowait, irq, softrig, steal,
guest, guest_nice = cpu_line

            Idle = idle + iowait
            NonIdle = user + nice + system + irq + softrig + steal

            Total = Idle + NonIdle
            # update dictionary
            cpu_infos.update({cpu_id: {'total': Total, 'idle': Idle}})

        return cpu_infos

    def Getcpuload(self):
        '''
                CPU_Percentage=((Total-PrevTotal)-(Idle-PrevIdle))/(Total-
PrevTotal)

                '''
        start = Test.Getcpuinfo(self)
        # wait a second
        sleep(1)
        stop = Test.Getcpuinfo(self)

        cpu_load = OrderedDict()

        for cpu in start:
            Total = stop[cpu]['total']
            PrevTotal = start[cpu]['total']

            Idle = stop[cpu]['idle']
            PrevIdle = start[cpu]['idle']
            CPU_Percentage = ((Total - PrevTotal) - (Idle - PrevIdle)) / (
Total - PrevTotal) * 100
            cpu_load.update({cpu: CPU_Percentage})
        return cpu_load  # dictionary that contains value as [cpuid: usage]

    def Getmeminfo(self):
        meminfo = OrderedDict()
        totmem = "cat /proc/meminfo | grep MemTotal|awk '{print$2}' "
        memfree = "cat /proc/meminfo | grep MemFree|awk '{print$2}' "
        swapTotal = " cat /proc/meminfo | grep SwapTotal|awk '{print$2}'"
        SwapFree = "cat /proc/meminfo | grep SwapFree|awk '{print$2}'"

        totalmemory = subprocess.check_output(totmem, shell=True).rstrip()
        swapmemory = subprocess.check_output(swapTotal, shell=True).rstrip()
        memoryfree = subprocess.check_output(memfree, shell=True).rstrip()
        swapfree = subprocess.check_output(SwapFree, shell=True).rstrip()

        meminfo['total memory'] = float(totalmemory.strip())
        meminfo['memory used'] = float(totalmemory.strip()) - float(memoryfree.strip())
        meminfo['free memory'] = float(memoryfree.strip())
        meminfo['swap memory'] = float(swapmemory.strip())
        meminfo['swap used'] = float(swapmemory.strip())-float(swapfree.strip())

        return meminfo

        return meminfo

    def Getdiskinfo(self):
        diskdict = OrderedDict()
        command = "df  | grep dev| grep '/$'|awk '{print $2,$3,$4,$5}'"
        xx = subprocess.check_output(command, shell=True, universal_newlines=True).split()

        diskdict['size'] = float(xx[0])
        diskdict['used space'] = float(xx[1])
        diskdict['available'] = float(xx[2])
        xxx = str(xx[3])
        xxx = xxx[:-1]

        diskdict['percent used'] = float(xxx)
        return diskdict

    def Getcputemp(self):
        jj = []
        tempd = OrderedDict()
        cmd = 'sensors|grep temp1|cut -d \' \' -f9|tr -d \' C \''
        xx = subprocess.check_output(cmd, shell=True).splitlines()

        for i in xx:
            jj.append(float(i))

        for i in range(0, len(jj)):
            tempd['temp' + str(i + 1)] = float(jj[i])
        return tempd

    def Getpower(self):
        jj = []
        powdict = OrderedDict()
        cmd = "sensors|grep power1|awk '{print $2}'"
        xx = subprocess.check_output(cmd, shell=True).splitlines()
        for i in xx:
            jj.append(float(i))
        # print(xx)

        for i in range(0, len(jj)):
            powdict['power' + str(i)] = float(jj[i])
```

```python
        return powdict


hname='hostname '
hostname=subprocess.check_output(hname,shell=True).strip()
hostname=hostname.decode('utf-8')
hstname=str(hostname)

# set the filenames to write
cpufile='aviral_'+hstname+'_cpufile.csv'
memfile='aviral_'+hstname+'_memfile.csv'
diskfile='aviral_'+hstname+'_diskfile.csv'
tempfile='aviral_'+hstname+'_tempfile.csv'
powerfile='aviral_'+hstname+'_powerfile.csv'


#Set the variables for connecting to the iot service
broker = ""
topic = "iot-2/evt/status/fmt/json"
username = "use-token-auth"
password = "altomonitor" #auth-token
organization = "qmznxv" #org_id
deviceType = "sensors"
deviceid ="altotest"

topic = "iot-2/evt/status/fmt/json"

#Creating the client connection
#Set clientID and broker
clientID = "d:" + organization + ":" + deviceType + ":" + deviceid
broker = organization + ".messaging.internetofthings.ibmcloud.com"
mqttc = mqtt.Client(clientID)

#Set authentication values, if connecting to registered service
if username is not "" :
    mqttc.username_pw_set(username, password=password)
    mqttc.connect(host=broker, port=1883, keepalive=60)


mqttc.loop_start()

tst=Test()

aa = tst.Getcpuload()
time=str(datetime.datetime.now().replace(microsecond=0))
aa.update({'time':time})
aa.move_to_end('time', last=False)
aa.update({'hostname':hstname})
aa.move_to_end('hostname', last=False)

#print(aa)
bb = tst.Getmeminfo()
aa.update({'time':time})
aa.move_to_end('time', last=False)
aa.update({'hostname':hstname})
aa.move_to_end('hostname', last=False)

#print(bb)

cc = tst.Getdiskinfo()
cc.update({'time':time})
cc.move_to_end('time', last=False)
aa.update({'hostname':hstname})
aa.move_to_end('hostname', last=False)

#print(cc)

dd = tst.Getcputemp()
dd.update({'time':time})
dd.move_to_end('time', last=False)
aa.update({'hostname':hstname})
aa.move_to_end('hostname', last=False)

#print(dd)

ee = tst.Getpower()
ee.update({'time':time})
ee.move_to_end('time', last=False)
aa.update({'hostname':hstname})
aa.move_to_end('hostname', last=False)

#print(ee)

cpulist = []  # aa
memlist = []  # bb
dsklist = []  # cc
templist = []  # dd
powlist = []  # ee

for i,j in aa.items():
    cpulist.append(j)

for i,j in bb.items():
    memlist.append(j)

for i,j in cc.items():
    dsklist.append(j)

for i,j in dd.items():
    templist.append(j)

for i,j in ee.items():
    powlist.append(j)


#create different filestream to save different parameters
```

```
# a for cpu(cpufile) , b for memory(memfile) , c for disk(diskfile), d for
    temperature(tempfile),e for power(powfile)
with open(cpufile, 'a') as a:
        writer = csv.writer(a)
        writer.writerow(cpulist)
        a.close()

with open(memfile, 'a') as b:
        writer = csv.writer(b)
        writer.writerow(memlist)
        b.close()

with open(diskfile, 'a') as c:
        writer = csv.writer(c)
        writer.writerow(dsklist)
        c.close()

with open(tempfile, 'a') as d:
        writer = csv.writer(d)
        writer.writerow(templist)
        d.close()

with open(powerfile, 'a') as e:
        writer = csv.writer(e)
        writer.writerow(powlist)
        e.close()

mac=':'.join(re.findall('..', '%012x' % uuid.getnode()))
##publishing to bluemix cloud
msgaa = json.JSONEncoder().encode({"d" :aa,"device_id" :mac})
mqttc.publish(topic, payload=msgaa, qos=1, retain=False)

msgbb = json.JSONEncoder().encode({"d": bb, "device_id": mac})
mqttc.publish(topic, payload=msgbb, qos=1, retain=False)
msgcc = json.JSONEncoder().encode({"d": cc, "device_id": mac})
mqttc.publish(topic, payload=msgcc, qos=1, retain=False)
msgdd = json.JSONEncoder().encode({"d": dd, "device_id": mac})
mqttc.publish(topic, payload=msgdd, qos=1, retain=False)
msgee = json.JSONEncoder().encode({"d": ee, "device_id": mac})
mqttc.publish(topic, payload=msgee, qos=1, retain=False)

print("success")
```