

# Optimizing the Resource Utilization in Cloud Computing Environment with Autoscaling using Machine Learning Methods

MSc Research Project  
Cloud Computing

SriMadhan Shettihalli Anandreddy

Student ID: x21230064

School of Computing  
National College of Ireland

Supervisor: Punit Gupta

National College of Ireland  
Project Submission Sheet  
School of Computing



<b>Student Name:</b>	SriMadhan Shettihalli Anandreddy
<b>Student ID:</b>	x21230064
<b>Programme:</b>	Cloud Computing
<b>Year:</b>	2023
<b>Module:</b>	MSc Research Project
<b>Supervisor:</b>	Punit Gupta
<b>Submission Due Date:</b>	18/09/2023
<b>Project Title:</b>	Optimizing the Resource Utilization in Cloud Computing Environment with Autoscaling using Machine Learning Methods
<b>Word Count:</b>	2031
<b>Page Count:</b>	16

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

**ALL** internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

<b>Signature:</b>	Sri Madhan Shettihalli Anandreddy
<b>Date:</b>	18th September 2023

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:**

Attach a completed copy of this sheet to each project (including multiple copies).	<input type="checkbox"/>
<b>Attach a Moodle submission receipt of the online project submission</b> , to each project (including multiple copies).	<input type="checkbox"/>
<b>You must ensure that you retain a HARD COPY of the project</b> , both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

<b>Office Use Only</b>	
Signature:	
Date:	
Penalty Applied (if applicable):	

# Optimizing the Resource Utilization in Cloud Computing Environment with Autoscaling using Machine Learning Methods

SriMadhan Shettihalli Anandreddy  
x21230064

## 1 Introduction

The primary goal of this document is to provide the reader with a comprehensive manual for successfully installing, configuring, and executing the project. It serves as a step-by-step guide for effectively setting up and running the system from start to finish. The document covers detailed installation instructions for each module, including relevant configuration options and parameters that may need to be set. Next, it delves into the overall system architecture at a high level first, providing the reader with an understanding of the different components and how they fit together. With the system installed, the execution flow and workflows are described to give the reader a clear picture of how modules interact during operation. The guide Also includes troubleshooting advice for common errors and issues that may be encountered during installation or execution. Following the steps outlined in this end-to-end guide will equip the reader with the knowledge to smoothly deploy the project in a new environment.. It is highly recommended that this document be thoroughly read and understood by those looking to operate the system.

## 2 Prerequisites

Users should have a basic working knowledge of Amazon Cloud, Ubuntu, Python programming, and common machine learning algorithms before using this system. Basic proficiency in Ubuntu is required to be able to navigate the file system, install packages, and execute commands through the terminal. Users should be familiar with Ubuntu administration including managing software, users, and processes. A fundamental understanding of the Python programming language is also expected. Users should be comfortable with Python syntax, data structures, control flow, modules, and virtual environments. Additionally, users should have experience with standard machine learning algorithms and models, including concepts like training, testing, feature engineering, and performance evaluation. Exposure to libraries like Scikit-Learn for implementing machine learning in Python is beneficial.

While advanced expertise is not required, having basic Ubuntu system administration skills, working knowledge of Python programming, and grounding in core machine learning concepts and techniques will ensure users can effectively work with the system.

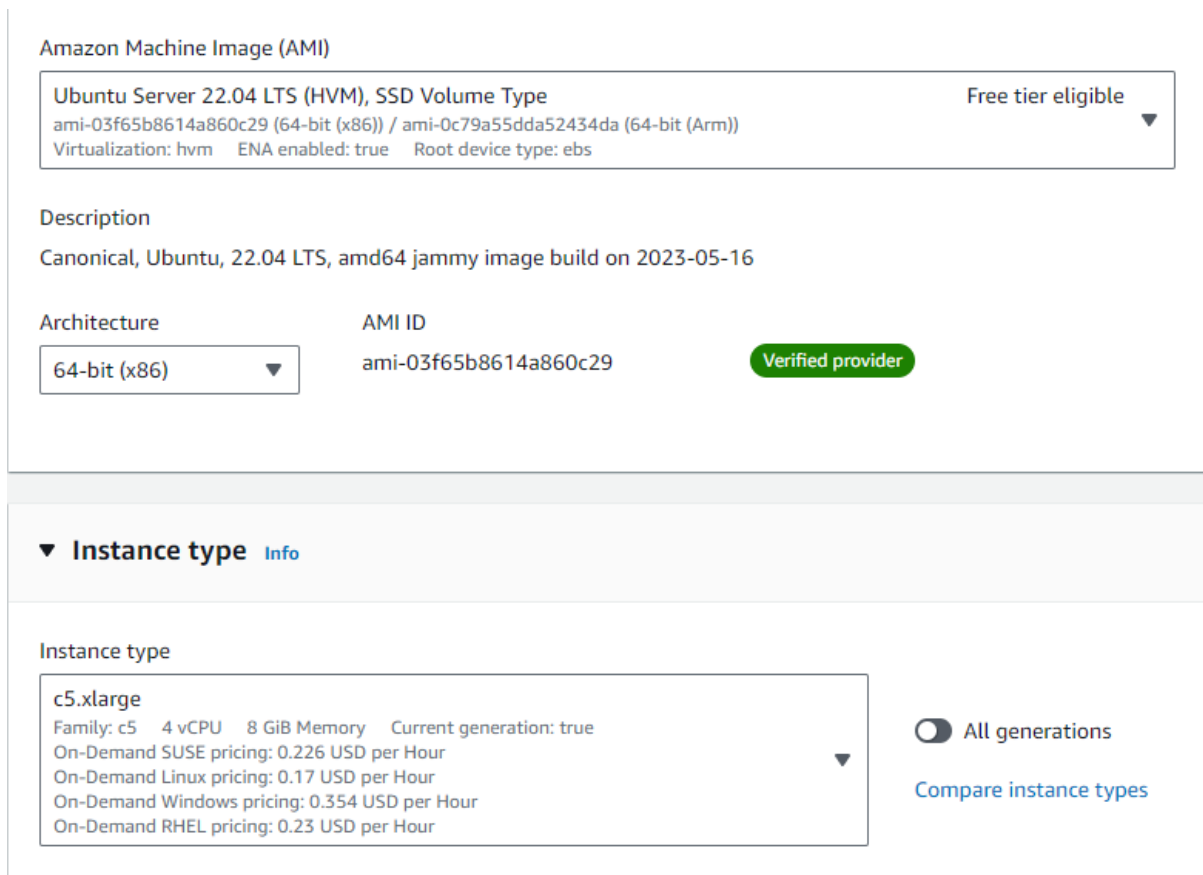


Figure 1: AWS EC2 Instance

### 3 Environment Setup

Figure 2 summarizes the hardware and software requirements for executing the project. I am using AWS cloud for my implementation. I have used configuration for EC2 instance as specified in Figure 1.

#### 3.1 Packages and Libraries

I have developed my project using python programming language and python version of 3.7 or later is required. You can install python latest version from <https://www.python.org/downloads/>

We are using multiple library functions in the project, we need pip package manager to install these library functions. pip can be installed by executing the below commands as shown in Figure 3 and Figure 4

```
sudo apt update && sudo apt upgrade
sudo apt install python3-pip
```

The library functions used in the project include TensorFlow for implementing deep learning models, scikit-learn for traditional machine learning algorithms, Matplotlib for visualization, NumPy for numerical processing, and Pandas for data analysis and manipulation. Boto3 provides interfaces for AWS services like accessing s3. The code is

Resource	Configuration
Operating System	Linux
Main Memory (RAM)	4GB
Number of CPU Cores	8 (Virtual Cores)
Storage	30GB
Programming Language	Python3
Python Libraries	Numpy, Pandas, Threading, Matplotlib, Sklearn, Keras, Tensorflow

Figure 2: Hardware and software Requirements

```

ubuntu@ip-172-31-47-25:~$ sudo apt update && sudo apt upgrade
Hit:1 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny InRelease
Hit:2 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates InRelease
Hit:3 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu janny-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
165 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
#
# You can verify the status of security fixes using the 'pro fix' command.
# E.g., a recent Ruby vulnerability can be checked with: 'pro fix USN-6219-1'
# For more detail see: https://ubuntu.com/security/notices/USN-6219-1
#
The following NEW packages will be installed:
linux-aws-5.19-headers-5.19.0-1029 linux-headers-5.19.0-1029-aws linux-lnag-5.19.0-1029-aws linux-modules-5.19.0-1029-aws
The following packages will be upgraded:
  amd64-microcode apport base-files bind9-dnswlts bind9-host bind9-libs binutils binutils-common binutils-x86-64-linux-gnu ca-certificates cloud-init curl distro-info dpkg ec2-hibinit-agent
  fwupd-signed gcc-12-base grub-common grub-pc grub-pc-bin grub2-common htagent intransfs-tools intransfs-tools-bin intransfs-tools-core iptables libbinutils libcap2 libcap2-bin libctf-nobfd libctf0
  libcurl3-gnutls libcurl4 libfwupd2 libfwupdplugins libgcc-s1 libgl1.0-0 libgl1.0-b0 libgl1.0-data libgpgme11 libip4tc2 libip6tc2 libldap-2.5-0 libldap-common libnss-glib2 libncurses6
  libncursesw6 libpam-cap libperl5.34 libpython3.10 libpython3.10-minimal libpython3.10-stdlib libssh-4 libssl3 libstdc++6 libstrfry libumtmd6 libx11-6 libx11-data libxtables12 linux-aws
  linux-headers-aws linux-headers-aws mdadm mokutil mtd-news-config ncurses-base ncurses-bin ncurses-term open-vn-tools openssl-client openssl-server openssh-client openssh-server openssh-sftp-server perl perl-base
  perl-modules-5.34 python-apt-common python3-apport python3-apt python3-debian python3-distro-info python3-distupgrade python3-distutils python3-gdbm python3-lib2to3 python3-problem-report
  python3-requests python3-software-properties python3.10 python3.10-minimal snapd software-properties-common tzdata ubuntu-advantage ubuntu-advantage-tools ubuntu-minimal ubuntu-release-upgrader-core ubuntu-server
  ubuntu-standard ufw vim vim-common vim-runtime vintiny xxd
165 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
60 standard LTS security updates
Need to get 128 MB of archives.
After this operation, 292 MB of additional disk space will be used.
Do you want to continue? [Y/n] yes
Get:1 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 mtd-news-config all 12ubuntu4.4 [4472 B]
Get:2 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 base-files amd64 12ubuntu4.4 [62.6 kB]
Get:3 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 dpkg amd64 1.21-ubuntu2.2 [1239 kB]
Get:4 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 ncurses-bin amd64 6.3-2ubuntu0.1 [184 kB]
Get:5 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 libperl5.34 amd64 5.34.0-3ubuntu1.2 [4818 kB]
Get:6 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 perl amd64 5.34.0-3ubuntu1.2 [232 kB]
Get:7 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 perl-base amd64 5.34.0-3ubuntu1.2 [1759 kB]
Get:8 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 perl-modules-5.34 all 5.34.0-3ubuntu1.2 [2977 kB]
Get:9 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 ncurses-base all 6.3-2ubuntu0.1 [20.2 kB]
Get:10 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 libpython3.10 amd64 3.10.12-1-22.04.2 [1949 kB]
Get:11 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 python3-distutils all 3.10.8-1-22.04 [129 kB]
Get:12 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 python3-lib2to3 all 3.10.8-1-22.04 [77.6 kB]
Get:13 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 libssl3 amd64 3.0.2-0ubuntu1.10 [1901 kB]
Get:14 http://us-west-2.ec2.archive.ubuntu.com/ubuntu janny-updates/main amd64 python3.10 amd64 3.10.12-1-22.04.2 [509 kB]

```

Figure 3: sudo apt update && sudo apt upgrade

developed in Python for its widespread use in machine learning and data science applications. TensorFlow enables building deep neural networks for approaches like CNNs and LSTM. Scikit-learn offers a range of classical ML algorithms like regression and random forests. Matplotlib and Pandas support analyzing and visualizing datasets. NumPy facilitates mathematical operations. Using these core libraries provides the modeling capabilities and data handling tools needed for an autoscaling prediction system based on machine learning. We can install this by executing the command as shown in Figure 5 : pip install scikit-learn tensorflow matplotlib

## 4 Methodology

### 4.1 Components

Project model consists of 5 main components, which are explained below.

- Data Generator: Generates simulated load data over time intervals. The load data

```
Invado operation python
ubuntu@ip-172-31-47-25:~$ sudo apt install python3-pip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  build-essential bzip2 cpp cpp-11 dpkg-dev fakeroot fontconfig-config fonts-dejavu-core g++ g++-11 gcc gcc-11 gcc-11-base javascript-common libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libasan libatomic1 libc-dev-bin libc-devtools libc6-dev libc-bin libcrypt-dev libdeflate0 libdpkg-perl libexpat1-dev libfakeroot libfile-fcntllock-perl libfontconfig1
  libgcc-11-dev libgdbm libgomp libisl23 libitm1 libjbig libjpeg-turbo libjpeg8 libjs-jquery libjs-sphinxdoc libjs-underscore libltdl7 libmpc3 libncurses-dev libpython3.10-dev libquadmath0
  libstdc++-11-dev libtiff5 libtirpc-dev libubsan libubsan1 libwebp libxpm4 linux-libc-dev lto-disabled-list make nanpages-dev python3-dev python3-pip python3-wheel python3.10-dev rpcsvc-proto zlib1g-dev
Suggested packages:
  bzip2-doc cpp-doc gcc-11-locales debconf-keyring g++-multilib g++-11-multilib gcc-11-doc gcc-multilib autoconf automake libtool flex bison gdb gcc-doc gcc-11-multilib apache2 | lighttpd | httpd
  glibc-doc bzr libgd-tools libstdc++-11-doc make-doc
The following NEW packages will be installed:
  build-essential bzip2 cpp cpp-11 dpkg-dev fakeroot fontconfig-config fonts-dejavu-core g++ g++-11 gcc gcc-11 gcc-11-base javascript-common libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libasan libatomic1 libc-dev-bin libc-devtools libc6-dev libc-bin libcrypt-dev libdeflate0 libdpkg-perl libexpat1-dev libfakeroot libfile-fcntllock-perl libfontconfig1
  libgcc-11-dev libgdbm libgomp libisl23 libitm1 libjbig libjpeg-turbo libjpeg8 libjs-jquery libjs-sphinxdoc libjs-underscore libltdl7 libmpc3 libncurses-dev libpython3.10-dev libquadmath0
  libstdc++-11-dev libtiff5 libtirpc-dev libubsan libubsan1 libwebp libxpm4 linux-libc-dev lto-disabled-list make nanpages-dev python3-dev python3-pip python3-wheel python3.10-dev rpcsvc-proto
  zlib1g-dev
0 upgraded, 64 newly installed, 0 to remove and 0 not upgraded.
Need to get 71.3 MB of archives.
After this operation, 239 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc-dev-bin amd64 2.35-0ubuntu3.1 [20.4 kB]
Get:2 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 linux-libc-dev amd64 5.15.0-78.85 [1307 kB]
Get:3 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 libcrypt-dev amd64 1:4.4.27-1 [112 kB]
Get:4 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 rpcsvc-proto amd64 1.4.2-0ubuntu2 [68.5 kB]
Get:5 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtirpc-dev amd64 1.3.0-2ubuntu1 [192 kB]
Get:6 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 libisl-dev amd64 1:3.0-2build1 [71.3 kB]
Get:7 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc6-dev amd64 2.35-0ubuntu3.1 [2099 kB]
Get:8 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 gcc-11-base amd64 11.4.0-1ubuntu1-22.04 [20.2 kB]
Get:9 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 libisl23 amd64 0.24-2build1 [727 kB]
Get:10 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 libmpc3 amd64 1:2.2-2build1 [46.9 kB]
Get:11 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 cpp-11 amd64 11.4.0-1ubuntu1-22.04 [10.0 MB]
Get:12 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 cpp amd64 4:11.2.0-1ubuntu1 [27.7 kB]
Get:13 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgcc-11 amd64 12.3.0-1ubuntu1-22.04 [48.3 kB]
Get:14 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgomp1 amd64 12.3.0-1ubuntu1-22.04 [126 kB]
Get:15 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libitm1 amd64 12.3.0-1ubuntu1-22.04 [30.2 kB]
Get:16 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libatomic1 amd64 12.3.0-1ubuntu1-22.04 [10.4 kB]
Get:17 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libasan amd64 11.4.0-1ubuntu1-22.04 [2282 kB]
Get:18 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libubsan amd64 12.3.0-1ubuntu1-22.04 [1069 kB]
Get:19 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtsan amd64 11.4.0-1ubuntu1-22.04 [2260 kB]
Get:20 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libubsan1 amd64 12.3.0-1ubuntu1-22.04 [976 kB]
Get:21 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libquadmath0 amd64 12.3.0-1ubuntu1-22.04 [154 kB]
Get:22 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgcc-11-dev amd64 11.4.0-1ubuntu1-22.04 [2517 kB]
Get:23 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 gcc-11 amd64 11.4.0-1ubuntu1-22.04 [20.1 MB]
Get:24 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 gcc amd64 4:11.2.0-1ubuntu1 [5112 B]
Get:25 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libstdc++-11-dev amd64 11.4.0-1ubuntu1-22.04 [2181 kB]
Get:26 http://us-west-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 g++-11 amd64 11.4.0-1ubuntu1-22.04 [11.4 MB]
```

Figure 4: sudo apt install python3-pip

```
ubuntu@ip-172-31-47-25:~$ pip install scikit-learn tensorflow matplotlib
Defaulting to user installation because normal site-packages is not writeable
Collecting scikit-learn
  Downloading scikit_learn-1.3.0-cp310-cp310-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (10.8 MB)
    ----- 10.8/10.8 MB 79.3 MB/s eta 0:00:00
Collecting tensorflow
  Downloading tensorflow-2.13.0-cp310-cp310-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (524.1 MB)
    ----- 224.1/524.1 MB 1.6 MB/s eta 0:00:00
Collecting matplotlib
  Downloading matplotlib-3.7.1-py3-none-any.whl (5.0 kB)
Collecting scipy==1.5.0
  Downloading scipy-1.11.1-cp310-cp310-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (36.3 MB)
    ----- 36.3/36.3 MB 38.9 MB/s eta 0:00:00
Collecting joblib>=1.1.1
  Downloading joblib-1.3.2-py3-none-any.whl (302 kB)
    ----- 302.2/302.2 kB 32.2 MB/s eta 0:00:00
Collecting threadpoolctl>=2.0.0
  Downloading threadpoolctl-3.2.0-py3-none-any.whl (15 kB)
Collecting numpy==1.17.3
  Downloading numpy-1.25.2-cp310-cp310-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (18.2 MB)
    ----- 18.2/18.2 MB 62.7 MB/s eta 0:00:00
Collecting gast<=0.4.0, >=0.2.1
  Downloading gast-0.4.0-py3-none-any.whl (9.8 kB)
Collecting wrapt==1.11.0
  Downloading wrapt-1.15.0-cp310-cp310-manylinux_2_5_x86_64_manylinux1_x86_64_manylinux2_17_x86_64_manylinux2014_x86_64.whl (78 kB)
    ----- 78.4/78.4 kB 18.6 MB/s eta 0:00:00
Collecting astunparse==1.6.0
  Downloading astunparse-1.6.3-py2.py3-none-any.whl (12 kB)
Collecting tensorflow-io-gcs-filesystem==0.23.1
  Downloading tensorflow_io_gcs_filesystem-0.33.0-cp310-cp310-manylinux_2_12_x86_64_manylinux2010_x86_64.whl (2.4 MB)
    ----- 2.4/2.4 MB 85.9 MB/s eta 0:00:00
Collecting tensorboard<2.14, >=2.13
  Downloading tensorboard-2.13.0-py3-none-any.whl (5.6 MB)
    ----- 5.6/5.6 MB 95.9 MB/s eta 0:00:00
Collecting tensorflow-estimator<2.14, >=2.13.0
  Downloading tensorflow_estimator-2.13.0-py2.py3-none-any.whl (440 kB)
    ----- 440.8/440.8 kB 46.4 MB/s eta 0:00:00
Collecting absl-py==1.0.0
  Downloading absl_py-1.4.0-py3-none-any.whl (126 kB)
    ----- 126.5/126.5 kB 29.4 MB/s eta 0:00:00
Collecting flatbuffers==23.1.21
  Downloading flatbuffers-23.5.26-py2.py3-none-any.whl (20 kB)
Requirement already satisfied: six>=1.12.0 in /usr/lib/python3/dist-packages (from tensorflow) (1.16.0)
Collecting protobuf<4.21.0, >=4.21.1, <4.21.2, >=4.21.3, <4.21.4, >=4.21.5, <5.0.0dev, >=3.20.3
  Downloading protobuf-4.24.0-cp37-abi3-manylinux2014_x86_64.whl (311 kB)
    ----- 311.6/311.6 kB 36.0 MB/s eta 0:00:00
Collecting libclang==13.0.0
  Downloading libclang-16.0.0-py2.py3-none-manylinux2010_x86_64.whl (22.9 MB)
```

Figure 5: pip install scikit-learn tensorflow matplotlib

```
sequence.py 2 X
sequence.py > ...
2 import numpy as np
3 import math
4 import random
5 import csv
6
7 SCALE = 1 # amplitude of the wave
8 VARIANCE = 0.1 # deviation from sine wave
9 INCREMENT = math.pi/36 #increment by this factor in each step
10
11 def generate_load_value(angle):
12     N = math.sin(angle) + (random.random()*VARIANCE)*SCALE
13     M = math.cos(angle)*(SCALE)
14     return abs(M) + abs(N)
15
16 if __name__ == "__main__":
17     angle = 0.1
18     x = []
19     y = []
20     for i in range(180):
21         angle += INCREMENT
22         x.append(angle)
23         y.append(generate_load_value(angle))
24
25     # Save the data to a CSV file
26     with open("generated_data.csv", "w", newline="") as csvfile:
27         csvwriter = csv.writer(csvfile)
28         csvwriter.writerow(["Angle", "Load Value"])
29         for angle, load_value in zip(x, y):
30             csvwriter.writerow([angle, load_value])
```

Figure 6: Sinusoidal Cloud workload generation

is a combination of sine and cosine wave with some randomness. Figure 6 shows code snippet for the same.

- **Task Generator:** This component is responsible for generating jobs. The Loader class adds generated jobs to the Scaler component and executes them. Code for handling this work is all included in jobs.py file.
- **Scaler:** Defines the Scaler class responsible for scaling the VM instance based on historical load data. It receives input from the Loader, maintains a history of load values, and uses a model to predict the next load value. It then adjusts the capacity of the executing node and executes the job. Code for handling this work is all included in scaler.py file.
- **Resource allocation unit:** this component is where actual tasks execute on. It defines the Node class that simulates a cloud instance capable of vertical scaling. The capacity of the resource allocation unit is adjusted by the Scaler. Code for handling this work is all included in node.py file. It keeps track of historical error and load data which will be utilized in comparing algorithms on the error metrics.
- **Visualizer:** With this component we plot graphs to visualize the data for better understanding. Code for handling this component is all written in visualizer.py file. It contains the Visualizer class responsible for generating various plots to visualize the project's performance. It creates plots related to load vs. capacity, underuse\_vs\_overuse, scaling error for different models, and delayed tasks for each model and stores these graphs in Amazon S3 bucket.

## 4.2 Algorithms

In this research we are using 3 different kinds of implementation for predicting auto-scaling in cloud.

### 4.2.1 Traditional threshold based method

In this method, scaling is done by a scaling factor which is predetermined by examining the historical data. Engineers analyze the load and use formula to calculate scaling factor. In our case we are calculating the scaling factor by taking the average of the historical load values. Figure 7 shows the code snippet for Rule-based model

### 4.2.2 Machine Learning Models

We are using two machine learning models for prediction of auto-scaling: Random Forest (RF) and Support Vector Regression (SVR) Model.

- **Support Vector Regression:** We are using a simple SVR model with linear kernel. this model works based on Simple linear boundary which is easier to interpret. It also has advantage of faster execution times when compared to Non-Linear patterns. Figure 8 shows the code snippet for SVR model.
- **Random Forest:** We are using simple Random forest model with shallow tree. The max\_depth parameter is set to 2 and random\_state is set to 0. Setting the random\_state to zero ensures consistent output results. After setting this parameters



```

class AvgModel:
    def predict(self, values):
        return sum(values)/len(values)

if __name__=="__main__":
    am = AvgModel()
    print(am.predict([1,2,3,4,5]))

```

Figure 7: Rule-based Model

```

from sklearn.svm import SVR
import numpy as np

class SVRModel:
    def predict(self, values):
        lr = SVR(kernel="linear")
        x = np.arange(len(values), dtype=np.float32).reshape([9,1])
        y = np.array(values, dtype=np.float32)
        lr.fit(x,y)
        x_p = np.array([[len(values)]], dtype=np.float32)
        predict = lr.predict(x_p).item()
        return predict

if __name__=="__main__":
    lr = SVRModel()
    print(lr.predict([1,2,3,4,5,6,7,8,9]))

```

Figure 8: Support Vector Regression (SVR) Model

```

from sklearn.ensemble import RandomForestRegressor
import numpy as np

class RFModel:
    def predict(self, values):
        lr = RandomForestRegressor(max_depth=2, random_state=0)
        x = np.arange(len(values), dtype=np.float32).reshape([9,1])
        y = np.array(values, dtype=np.float32)
        lr.fit(x,y)
        x_p = np.array([[len(values)]], dtype=np.float32)
        predict = lr.predict(x_p).item()
        return predict

if __name__=="__main__":
    lr = RFModel()
    print(lr.predict([1,2,3,4,5,6,7,8,9]))

```

Figure 9: Random Forest Model

we are reshaping the input values into a 2D array with one column as required by the model and train the model on these input values. Figure 9 shows the code snippet for SVR model.

### 4.2.3 Deep Learning Models

We are predicting auto-scaling using two deep learning models: Gated recurrent unit (GRU) and an ensemble model- CNN\_LSTM Model.

- CNN\_LSTM Model: The primary concept behind using these models on time-series data is that while CNN models are beneficial in extracting the valuable features and may filter out the noise of the input data, LSTM networks are capable of capturing sequence pattern information. The model workflow is explained as below: Input data is loaded and reshaped to 3d array format as samples, time steps and features. A sequential model is then created and convolution model is applied for feature extraction. then LSTM is applied for sequence modelling. Finally a dense layer is added to prediction output. We are using Adam optimizer with learning rate set to 0.0001. also loss parameter is set to mean squared error. then the model is trained for 1000 epochs with batch size of 128. after 1000 epochs, the loss value remains constant. Figure 10 shows the training for CNN\_LSTM model.
- Gated recurrent unit Model: the workflow of model is as follows: The input data is loaded and it contains the features x and targets y. A Sequential Keras model is created. A GRU layer is added to the model to handle the sequential nature of the data. The input shape is set to (9,1) based on the 9 time steps. A Dense output layer is added for making a single value prediction. The model is compiled with

```
model.fit(x,y,epochs=1000,batch_size=128)
Python
Epoch 1/1000
2023-08-04 00:06:29.790287: I tensorflow/core/common_runtime/executor.cc:1197 [/device:CPU:0] (DEBUG INFO) Executor s
[[{{node gradients/split_2_grad/concat/split_2/split_dim}}]]
2023-08-04 00:06:29.791213: I tensorflow/core/common_runtime/executor.cc:1197 [/device:CPU:0] (DEBUG INFO) Executor s
[[{{node gradients/split_grad/concat/split/split_dim}}]]
2023-08-04 00:06:29.791860: I tensorflow/core/common_runtime/executor.cc:1197 [/device:CPU:0] (DEBUG INFO) Executor s
[[{{node gradients/split_1_grad/concat/split_1/split_dim}}]]
2023-08-04 00:06:30.168644: I tensorflow/core/common_runtime/executor.cc:1197 [/device:CPU:0] (DEBUG INFO) Executor s
[[{{node gradients/split_2_grad/concat/split_2/split_dim}}]]
2023-08-04 00:06:30.169509: I tensorflow/core/common_runtime/executor.cc:1197 [/device:CPU:0] (DEBUG INFO) Executor s
[[{{node gradients/split_grad/concat/split/split_dim}}]]
2023-08-04 00:06:30.170147: I tensorflow/core/common_runtime/executor.cc:1197 [/device:CPU:0] (DEBUG INFO) Executor s
[[{{node gradients/split_1_grad/concat/split_1/split_dim}}]]
2023-08-04 00:06:31.602249: I tensorflow/compiler/xla/service/service.cc:169] XLA service 0x7ef99813f260 initialized f
2023-08-04 00:06:31.602269: I tensorflow/compiler/xla/service/service.cc:177] StreamExecutor device (0): NVIDIA Gefo
2023-08-04 00:06:31.632212: I tensorflow/compiler/mlir/tensorflow/utils/dump_mlir_util.cc:269] disabling MLIR crash re
2023-08-04 00:06:31.887443: I ./tensorflow/compiler/jit/device\_compiler.h:180] Compiled cluster using XLA! This line
1/1 [=====] - 3s 3s/step - loss: 1.0846
Epoch 2/1000
1/1 [=====] - 0s 5ms/step - loss: 1.0699
Epoch 3/1000
1/1 [=====] - 0s 4ms/step - loss: 1.0553
Epoch 4/1000
1/1 [=====] - 0s 5ms/step - loss: 1.0408
Epoch 5/1000
1/1 [=====] - 0s 4ms/step - loss: 1.0265
Epoch 6/1000
1/1 [=====] - 0s 4ms/step - loss: 1.0122
Epoch 7/1000
1/1 [=====] - 0s 4ms/step - loss: 0.9980
Epoch 8/1000
1/1 [=====] - 0s 4ms/step - loss: 0.9840
Epoch 9/1000
1/1 [=====] - 0s 4ms/step - loss: 0.9700
Epoch 10/1000
1/1 [=====] - 0s 6ms/step - loss: 0.9562
Epoch 11/1000
1/1 [=====] - 0s 4ms/step - loss: 0.9425
Epoch 12/1000
1/1 [=====] - 0s 4ms/step - loss: 0.9288
Epoch 13/1000
1/1 [=====] - 0s 3ms/step - loss: 0.9153
...
Epoch 999/1000
1/1 [=====] - 0s 3ms/step - loss: 9.1205e-04
Epoch 1000/1000
1/1 [=====] - 0s 4ms/step - loss: 9.1205e-04
```

Figure 10: CNN\_LSTM Model

```

model_fit(x,y,epochs=100) Python
Epoch 1/100
2023-08-04 00:11:43.036718: I tensorflow/core/common_runtime/executor.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an error and you can ignore this message): IN
[[[{}(node_gradients/split_2_grad/concat/split_2/split_dim)]]]
2023-08-04 00:11:43.037582: I tensorflow/core/common_runtime/executor.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an error and you can ignore this message): IN
[[[{}(node_gradients/split_grad/concat/split/split_dim)]]]
2023-08-04 00:11:43.038216: I tensorflow/core/common_runtime/executor.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an error and you can ignore this message): IN
[[[{}(node_gradients/split_1_grad/concat/split_1/split_dim)]]]
2023-08-04 00:11:43.037719: I tensorflow/core/common_runtime/executor.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an error and you can ignore this message): IN
[[[{}(node_gradients/split_2_grad/concat/split_2/split_dim)]]]
2023-08-04 00:11:43.038554: I tensorflow/core/common_runtime/executor.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an error and you can ignore this message): IN
[[[{}(node_gradients/split_grad/concat/split/split_dim)]]]
2023-08-04 00:11:43.039185: I tensorflow/core/common_runtime/executor.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an error and you can ignore this message): IN
[[[{}(node_gradients/split_1_grad/concat/split_1/split_dim)]]]
2023-08-04 00:11:44.226833: I tensorflow/compiler/xla/stream_executor/cuda/cuda_dnn.cc:424] Loaded cuDNN version 8600
2023-08-04 00:11:44.302711: I tensorflow/compiler/xla/stream_executor/cuda/cuda_blas.cc:637] TensorFlow-32 will be used for the matrix multiplication. This will only be logged once.
2023-08-04 00:11:44.324167: I tensorflow/compiler/xla/service/service.cc:169] XLA service 0x7f42d0131370 initialized for platform CUDA (this does not guarantee that XLA will be used). Devices:
2023-08-04 00:11:44.324193: I tensorflow/compiler/xla/service/service.cc:177] StreamExecutor device (0): NVIDIA GeForce RTX 3080 Ti, Compute Capability 8.6
2023-08-04 00:11:44.327109: I tensorflow/compiler/mlir/tensorflow/utils/dump_mlir_util.cc:269] disabling MLIR crash reproducer, set env var 'MLIR_CRASH_REPRODUCER_DIRECTORY' to enable.
2023-08-04 00:11:44.417498: I tensorflow/compiler/jit/device_compiler_b.cc:180] Compiled cluster using XLA! This line is logged at most once for the lifetime of the process.
4/4 [=====] - 2s 4ms/step - loss: 1.1192
Epoch 2/100
4/4 [=====] - 0s 2ms/step - loss: 0.6746
Epoch 3/100
4/4 [=====] - 0s 3ms/step - loss: 0.3251
Epoch 4/100
4/4 [=====] - 0s 3ms/step - loss: 0.0892
Epoch 5/100
4/4 [=====] - 0s 2ms/step - loss: 0.0056
Epoch 6/100
4/4 [=====] - 0s 3ms/step - loss: 0.0497
Epoch 7/100
4/4 [=====] - 0s 2ms/step - loss: 0.0477
Epoch 8/100
4/4 [=====] - 0s 3ms/step - loss: 0.0114
Epoch 9/100
4/4 [=====] - 0s 3ms/step - loss: 0.0015
Epoch 10/100
4/4 [=====] - 0s 3ms/step - loss: 0.0070
Epoch 11/100
4/4 [=====] - 0s 3ms/step - loss: 0.0094
Epoch 12/100
4/4 [=====] - 0s 3ms/step - loss: 0.0063
Epoch 13/100
4/4 [=====] - 0s 3ms/step - loss: 0.0023
...
Epoch 99/100
4/4 [=====] - 0s 3ms/step - loss: 9.4643e-04
Epoch 100/100
4/4 [=====] - 0s 2ms/step - loss: 9.5691e-04
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```

Figure 11: GRU Model

Adam optimizer using a learning rate of 0.001 and MSE loss function. The model is trained for 100 epochs by fitting on the input x and target y data. The trained GRU model is then saved. Figure 11 shows the GRU model training for 100 epochs.

## 5 Execution and Results

### 5.0.1 Execution steps

Once we have Environment setup and all the project files copied to destination path, we are now good to execute the models. We have created python files for invoking all the components specified in project methodology for each Model.

- Data generation: Code for generating data is stored in sequence.py python file. Executing this file creates ds.pkl file which will be used for training ML and deep learning algorithms. Also a sample graph is plotted to visualize the generated data. Figure 12 shows sample plot of dataset generated using sequence.py Figure 13 shows the command for generating synthetic data. We can see the output file ds.pkl created after executing the python file.
- Train Algorithms: We use the synthetic dataset generated in above step to train Moving\_avg, RF, SVR, GRU, CNN\_LSTM algorithms. Figure 14, Figure 15 and Figure 16 shows the output generated for Model training. For moving\_avg we had fed input as [1,2,3,4,5] resulting in average of 3 as shown in Figure 14. We could see different predicted capacity for different models.

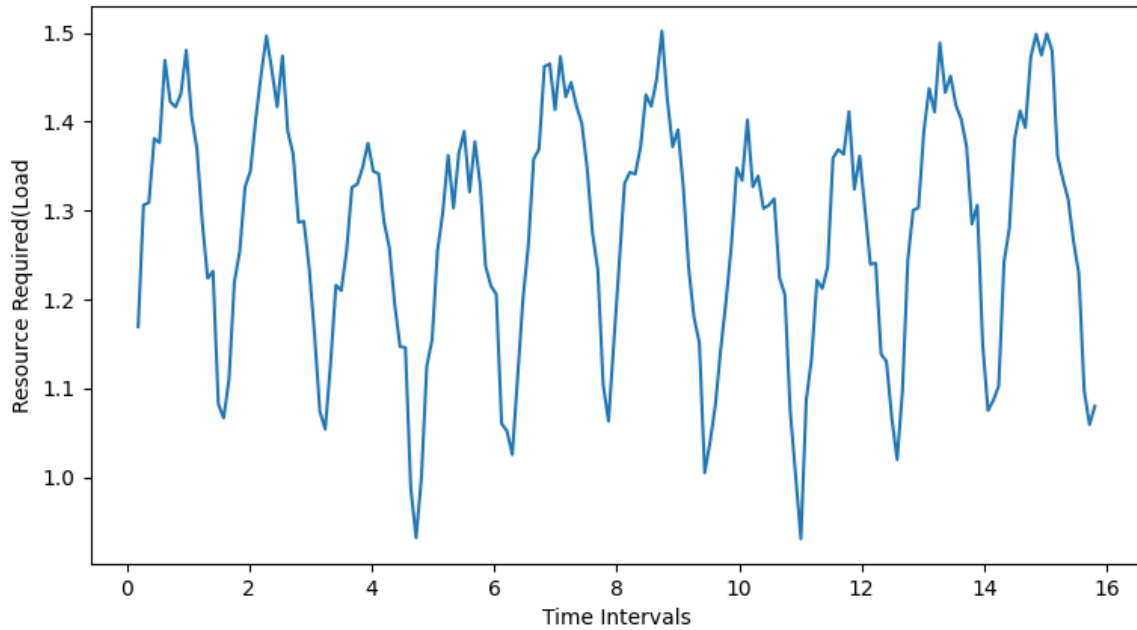


Figure 12: Dataset graphical representation

```

ubuntu@ip-172-31-47-25:/home/autoscaling/x21230064$ python3 sequence.py
ubuntu@ip-172-31-47-25:/home/autoscaling/x21230064$ ls -lrth
total 512K
-rw-rw-r-- 1 ubuntu ubuntu 43K Aug 9 20:03 gru_notebook.ipynb
-rw-rw-r-- 1 ubuntu ubuntu 1.3K Aug 9 20:03 y.csv
-rw-rw-r-- 1 ubuntu ubuntu 9.0K Aug 9 20:03 x.csv
drwxrwxr-x 3 ubuntu ubuntu 4.0K Aug 10 10:54 gru
drwxrwxr-x 3 ubuntu ubuntu 4.0K Aug 10 10:54 cnn_lstm
-rw-rw-r-- 1 ubuntu ubuntu 134K Aug 10 11:41 cnn_lstm.ipynb
-rw-rw-r-- 1 ubuntu ubuntu 11K Aug 10 13:57 ds.csv
-rw-rw-r-- 1 ubuntu ubuntu 1012 Aug 11 15:21 scaler.py
-rw-rw-r-- 1 ubuntu ubuntu 1.4K Aug 11 15:23 node.py
-rw-rw-r-- 1 ubuntu ubuntu 667 Aug 11 15:44 cnn_lstm.py
-rw-rw-r-- 1 ubuntu ubuntu 878 Aug 11 15:44 Execute_CNN_LSTM.py
-rw-rw-r-- 1 ubuntu ubuntu 860 Aug 11 15:44 Execute_GRU.py
-rw-rw-r-- 1 ubuntu ubuntu 892 Aug 11 15:45 Execute_moving_avg.py
-rw-rw-r-- 1 ubuntu ubuntu 886 Aug 11 15:45 Execute_RF.py
-rw-rw-r-- 1 ubuntu ubuntu 882 Aug 11 15:46 Execute_SVR.py
-rw-rw-r-- 1 ubuntu ubuntu 660 Aug 11 15:46 gru_model.py
-rw-rw-r-- 1 ubuntu ubuntu 1.9K Aug 11 15:46 job.py
-rw-rw-r-- 1 ubuntu ubuntu 237 Aug 11 15:47 mean_model.py
-rw-rw-r-- 1 ubuntu ubuntu 547 Aug 11 15:48 rf_model.py
-rw-rw-r-- 1 ubuntu ubuntu 508 Aug 11 18:02 svr_model.py
-rw-rw-r-- 1 ubuntu ubuntu 922 Aug 12 13:17 sequence.py
-rw-rw-r-- 1 ubuntu ubuntu 382 Aug 12 13:18 test.py
-rw-rw-r-- 1 ubuntu ubuntu 27K Aug 12 21:36 cnn_lstm.pkl
-rw-rw-r-- 1 ubuntu ubuntu 27K Aug 12 21:37 gru.pkl
-rw-rw-r-- 1 ubuntu ubuntu 27K Aug 12 21:40 rf.pkl
-rw-rw-r-- 1 ubuntu ubuntu 27K Aug 12 21:40 svr.pkl
-rw-rw-r-- 1 ubuntu ubuntu 27K Aug 12 21:40 avg.pkl

```

Figure 13: Data Generation

```
ubuntu@ip-172-31-47-25:/home/autoscaling/x21230064$ python3 mean_model.py
3.0
ubuntu@ip-172-31-47-25:/home/autoscaling/x21230064$ python3 rf_model.py
8.2221666666666668
ubuntu@ip-172-31-47-25:/home/autoscaling/x21230064$ python3 svr_model.py
9.875
```

Figure 14: Moving\_avg, RF, SVR Model Training

```
2023-08-13 16:01:00.006523: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-08-13 16:01:00.051505: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-08-13 16:01:00.051827: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in
performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-08-13 16:01:00.672685: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
1/1 [=====] - 0s 229ms/step
1.0677597522735596
1.0150383
```

Figure 15: GRU Model Training

- Execute Trained Models on dynamically generated loads: code for this step are included in Execute\_Model.py files. Executing these files creates jobs with random loads and the scalar component loads the respective algorithm in the file and predicts the capacity required- the scaling factor to which the resource allocation unit vertically scales the node. visualizer plots graphs for the log data generated by resource allocation unit.

Executing these files creates jobs with random loads and the scalar component loads the respective Model and predicts the capacity required - the scaling factor to which the resource allocation unit scales the node. visualizer plots graphs for the log data generated by resource allocation unit. Figure 17, Figure 18, Figure 8, Figure 20, Figure 21 shows the sample output generated when executing these files for Moving\_avg, RF, SVR, GRU, CNN\_LSTM models respectively.

We Could see in figures that jobs are first loaded and scalar component predicts the required capacity, then the node is scaled to the predicted capacity. Once the task is assigned to the node, if the predicted capacity is less than the required capacity to execute the task, the task has to wait until the node has capacity to execute the job. We can see in the same in sample outputs for few cycles the job executes as soon as it is assigned to node. But for few some cycles, job is delayed for a while.

The visualizer component generates graphs and uploads the same to Amazon S3 bucket. Figure 22 shows the graphs generated and stored in AWS S3 accounts.

```
2023-08-13 16:00:05.322343: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-08-13 16:00:05.577355: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-08-13 16:00:05.578276: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in
performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-08-13 16:00:06.527551: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
1/1 [=====] - 0s 291ms/step
1.0531905889511108
1.0150383
```

Figure 16: CNN\_LSTM Model Training

```

ubuntu@ip-172-31-47-25:/home/autoscaling/x21230064$ python3 Execute_moving_avg.py
Added job with load value: 1.030584475237713, Cycle: 1
Added job with load value: 1.1572847978126335, Cycle: 2
Added job with load value: 1.1875535753966915, Cycle: 3
Added job with load value: 1.2407194973869002, Cycle: 4
Added job with load value: 1.37649170336611, Cycle: 5
Added job with load value: 1.375301708226373, Cycle: 6
Added job with load value: 1.4425436106910472, Cycle: 7
Added job with load value: 1.4325192683821664, Cycle: 8
Added job with load value: 1.4769519461139733, Cycle: 9
Loader is executing
Added job with load value: 1.506946409983137, Cycle: 10
Executing scaler for job load: 1.506946409983137
Predicted capacity: 1.3551458352621146
Node capacity scaled to: 1.3551458352621146
Job with load factor 1.506946409983137 delayed due to resource underuse.
Job executed on node with load factor: 1.506946409983137
Scaler executed for job load: 1.506946409983137

Added job with load value: 1.479360659768233, Cycle: 11
Executing scaler for job load: 1.479360659768233
Predicted capacity: 1.39093204214607
Node capacity scaled to: 1.39093204214607
Job with load factor 1.479360659768233 delayed due to resource underuse.
Job executed on node with load factor: 1.479360659768233
Scaler executed for job load: 1.479360659768233

```

Figure 17: Moving\_avg Model Execution on dynamically generated task

```

ubuntu@ip-172-31-47-25:/home/autoscaling/x21230064$ python3 Execute_RF.py
Added job with load value: 1.0063207137354588, Cycle: 1
Added job with load value: 1.1459475064867468, Cycle: 2
Added job with load value: 1.250955334288287, Cycle: 3
Added job with load value: 1.3187432441713167, Cycle: 4
Added job with load value: 1.304809387990353, Cycle: 5
Added job with load value: 1.335183161589374, Cycle: 6
Added job with load value: 1.413942472304122, Cycle: 7
Added job with load value: 1.4149798436068137, Cycle: 8
Added job with load value: 1.4369470250929077, Cycle: 9
Loader is executing
Added job with load value: 1.4173617469580866, Cycle: 10
Executing scaler for job load: 1.4173617469580866
Predicted capacity: 1.4235707680185639
Node capacity scaled to: 1.4235707680185639
Job with load factor 1.4173617469580866 executed successfully.
Job executed on node with load factor: 1.4173617469580866
Scaler executed for job load: 1.4173617469580866

Added job with load value: 1.449068323229044, Cycle: 11
Executing scaler for job load: 1.449068323229044
Predicted capacity: 1.4395852833986282
Node capacity scaled to: 1.4395852833986282
Job with load factor 1.449068323229044 delayed due to resource underuse.
Job executed on node with load factor: 1.449068323229044
Scaler executed for job load: 1.449068323229044

```

Figure 18: RF Model Execution on dynamically generated task

```
ubuntu@ip-172-31-47-25:/home/autoscaling/x21230064$ python3 Execute_SVR.py
Added job with load value: 1.090641409527635, Cycle: 1
Added job with load value: 1.1564025346174727, Cycle: 2
Added job with load value: 1.2243906297162588, Cycle: 3
Added job with load value: 1.2528731800256052, Cycle: 4
Added job with load value: 1.3305177336640515, Cycle: 5
Added job with load value: 1.3898710359210895, Cycle: 6
Added job with load value: 1.3756924374674924, Cycle: 7
Added job with load value: 1.3978521285231036, Cycle: 8
Added job with load value: 1.4090229685368048, Cycle: 9
Loader is executing
Added job with load value: 1.480489772010784, Cycle: 10
Executing scaler for job load: 1.480489772010784
Predicted capacity: 1.396000623703003
Node capacity scaled to: 1.396000623703003
Job with load factor 1.480489772010784 delayed due to resource underuse.
Job executed on node with load factor: 1.480489772010784
Scaler executed for job load: 1.480489772010784

Added job with load value: 1.4672953648927152, Cycle: 11
Executing scaler for job load: 1.4672953648927152
Predicted capacity: 1.3965180465153284
Node capacity scaled to: 1.3965180465153284
Job with load factor 1.4672953648927152 delayed due to resource underuse.
Job executed on node with load factor: 1.4672953648927152
Scaler executed for job load: 1.4672953648927152

Added job with load value: 1.4231265336625583, Cycle: 12
Executing scaler for job load: 1.4231265336625583
Predicted capacity: 1.3942980051040648
Node capacity scaled to: 1.3942980051040648
Job with load factor 1.4231265336625583 delayed due to resource underuse.
Job executed on node with load factor: 1.4231265336625583
Scaler executed for job load: 1.4231265336625583

Added job with load value: 1.379312191179126, Cycle: 13
Executing scaler for job load: 1.379312191179126
Predicted capacity: 1.405503749847412
Node capacity scaled to: 1.405503749847412
Job with load factor 1.379312191179126 executed successfully.
Job executed on node with load factor: 1.379312191179126
Scaler executed for job load: 1.379312191179126
```

Figure 19: SVR Model Execution on dynamically generated task



```

ubuntu@ip-172-31-47-25: /home/autoscaling/x21230064$ python3 Execute_GRU.py
2023-08-13 15:05:56.892496: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-08-13 15:05:56.934962: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-08-13 15:05:56.935511: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
To enable the following instructions: AVX2 AVX512F FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-08-13 15:05:57.591989: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
Added job with load value: 1.024985917056756, Cycle: 1
Added job with load value: 1.1018388620205226, Cycle: 2
Added job with load value: 1.170248221082432, Cycle: 3
Added job with load value: 1.2664077542063412, Cycle: 4
Added job with load value: 1.359787691068134, Cycle: 5
Added job with load value: 1.4192783214611135, Cycle: 6
Added job with load value: 1.4277463833176602, Cycle: 7
Added job with load value: 1.4682001105010225, Cycle: 8
Added job with load value: 1.4295642162978848, Cycle: 9
Loader is executing
Added job with load value: 1.4466094958102445, Cycle: 10
Executing scaler for job load: 1.4466094958102445
1/1 [=====] - 0s 268ms/step
Predicted capacity: 1.2998627424240112
Node capacity scaled to: 1.2998627424240112
Job with load factor 1.4466094958102445 delayed due to resource underuse.
Job executed on node with load factor: 1.4466094958102445
Scaler executed for job load: 1.4466094958102445

Added job with load value: 1.4586248409831564, Cycle: 11
Executing scaler for job load: 1.4586248409831564
1/1 [=====] - 0s 14ms/step
Predicted capacity: 1.3109842644500732
Node capacity scaled to: 1.3109842644500732
Job with load factor 1.4586248409831564 delayed due to resource underuse.
Job executed on node with load factor: 1.4586248409831564
Scaler executed for job load: 1.4586248409831564

```

Figure 20: GRU Model Execution on dynamically generated task

```

ubuntu@ip-172-31-47-25: /home/autoscaling/x21230064$ python3 Execute_CNN_LSTM.py
2023-08-13 15:06:52.575837: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-08-13 15:06:52.618903: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-08-13 15:06:52.619414: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
To enable the following instructions: AVX2 AVX512F FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-08-13 15:06:53.270231: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
Added job with load value: 1.0186749131653979, Cycle: 1
Added job with load value: 1.1762775880978076, Cycle: 2
Added job with load value: 1.2492608814842479, Cycle: 3
Added job with load value: 1.3076002905588963, Cycle: 4
Added job with load value: 1.3446671679223919, Cycle: 5
Added job with load value: 1.4086084293436524, Cycle: 6
Added job with load value: 1.428285878248388, Cycle: 7
Added job with load value: 1.4913838254389118, Cycle: 8
Added job with load value: 1.4232525626095347, Cycle: 9
Loader is executing
Added job with load value: 1.4662550623645054, Cycle: 10
Executing scaler for job load: 1.4662550623645054
1/1 [=====] - 0s 336ms/step
Predicted capacity: 1.3218523263931274
Node capacity scaled to: 1.3218523263931274
Job with load factor 1.4662550623645054 delayed due to resource underuse.
Job executed on node with load factor: 1.4662550623645054
Scaler executed for job load: 1.4662550623645054

Added job with load value: 1.4165249216023408, Cycle: 11
Executing scaler for job load: 1.4165249216023408
1/1 [=====] - 0s 13ms/step
Predicted capacity: 1.3456840515136719
Node capacity scaled to: 1.3456840515136719
Job with load factor 1.4165249216023408 delayed due to resource underuse.
Job executed on node with load factor: 1.4165249216023408
Scaler executed for job load: 1.4165249216023408

```

Figure 21: CNN\_LSTM Model Execution on dynamically generated task

Amazon S3 > Buckets > x21230064

**x21230064** Info

Publicly accessible

Objects Properties Permissions Metrics Management Access Points

Objects (12)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 Inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

Find objects by prefix

Name	Type	Last modified	Size	Storage class
delay_bar_chart.png	png	August 13, 2023, 15:45:36 (UTC+01:00)	28.9 KB	Standard
error_bar_chart.png	png	August 13, 2023, 15:45:36 (UTC+01:00)	29.0 KB	Standard
Load_vs_CapacityCNN_LSTM.png	png	August 13, 2023, 16:08:26 (UTC+01:00)	45.6 KB	Standard
Load_vs_CapacityGRU.png	png	August 13, 2023, 16:09:12 (UTC+01:00)	45.1 KB	Standard
Load_vs_Capacitymoving avg.png	png	August 13, 2023, 16:07:37 (UTC+01:00)	44.7 KB	Standard
Load_vs_Capacityrandom forest ensemble.png	png	August 13, 2023, 16:11:14 (UTC+01:00)	48.1 KB	Standard
Load_vs_CapacitySVR.png	png	August 13, 2023, 14:48:52 (UTC+01:00)	48.6 KB	Standard
Overused_vs_Underused_CNN_LSTM.png	png	August 13, 2023, 16:08:27 (UTC+01:00)	38.7 KB	Standard
Overused_vs_Underused_GRU.png	png	August 13, 2023, 16:09:13 (UTC+01:00)	42.0 KB	Standard
Overused_vs_Underused_moving avg.png	png	August 13, 2023, 16:07:38 (UTC+01:00)	38.9 KB	Standard
Overused_vs_Underused_random forest ensemble.png	png	August 13, 2023, 16:11:14 (UTC+01:00)	45.1 KB	Standard
Overused_vs_Underused_SVR.png	png	August 13, 2023, 14:48:53 (UTC+01:00)	40.0 KB	Standard

Figure 22: Graphs stored in AWS S3 bucket

## 5.1 Evaluation Metrics

We have used the following evaluation metrics to compare models performance in predicting auto-scaling:

- **Overused\_vs\_Underused:** this graph represents whether the predicted capacity for executing the job resulted in over usage or an under utilization of resource.
- **Combined error:** this is a bar chart representation showing the combined error for different models. Error history from different model instances are retrieved and sum of absolute error values are then plotted.
- **Delayed\_tasks:** This is a graph showing the number of delayed tasks for different models. It retrieves the error history from different model instances and counts the number of delayed tasks (where error  $\neq 0$ ) for each model. The number of delayed tasks for different models is then plotted using matplotlib as a bar chart.
- **Load vs Capacity:** This graph represents the actual load versus predicted load required for executing the task for each model.