National College of Ireland

Configuration Manual

MSc Research Project
Master of Science in Cloud Computing

Hima Shree Manyam Sunil
Student ID: X23189851

School of Computing
National College of Ireland

Supervisor: Shreyas Setlur Arun

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Hima Shree Manyam Sunil
Student ID:	X23189851
Programme:	MSc in Cloud computing **Year:** 2024/2025
Module:	MSc Research Project
Lecturer:	Shreyas Setlur Arun
Submission Due Date:	12/12/2024
Project Title:	Performance Evaluation of AWS and Azure Cloud Platforms Using TensorFlow Framework
Word Count:	723 **Page Count:** 6

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.

Signature:	Hima Shree Manyam Sunil
Date:	12-12-24

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

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

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

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Hima Shree Manyam Sunil
x23189851@student.ncirl.ie

1 Introduction

This manual provides detailed guidelines for setting up, executing, and evaluating experiments for benchmarking the performance of AWS EC2 and Azure VM platforms using TensorFlow for deep learning tasks. The evaluation focuses on the following metrics:

- CPU utilization
- Memory consumption
- Disk I/O performance
- Training time
- Cost efficiency
- Model accuracy

This study involves training the GoogleNet model on both platforms with identical configurations to ensure fair comparisons. The results of this study aim to aid researchers and practitioners in selecting the optimal platform for their machine learning workloads.

2 System Requirements

2.1 Hardware Requirements

- **CPU**: Minimum 8 cores
- **Memory**: At least 32 GB RAM
- **Storage**: High-throughput SSD storage
- **Internet**: High-speed connection for accessing cloud platforms

2.2 Software Requirements

- **Operating System**: Windows Server 2022 or equivalent
- **Programming Language**: Python 3.9 or higher
- **Deep Learning Framework**: TensorFlow 2.12
- **Monitoring Tools**: AWS CloudWatch and Azure Monitor

- **Version Control**: Git for tracking code changes

2.3 Dataset

- **Dataset Name:** Stanford University Elbow X-ray Dataset
- **Dataset Size:** Approximately 4300 images
- **Task:** Binary classification (Positive/Negative)

2.4 Preprocessing Techniques:

- **Image Resizing:** Convert images to 224x224 pixels using OpenCV**.**
- **Normalization:** Scale pixel values to the range [0, 1].
- **Augmentation:** Apply rotation, zoom, shear, and flips using TensorFlow's ImageDataGenerator**.**

3. Cloud Platform Setup

3.1 AWS EC2 Setup

1. First create an AWS account and sign in to the AWS Console and go to the EC2 Dashboard to create a EC2 first click on launch instance as shown in the figure.

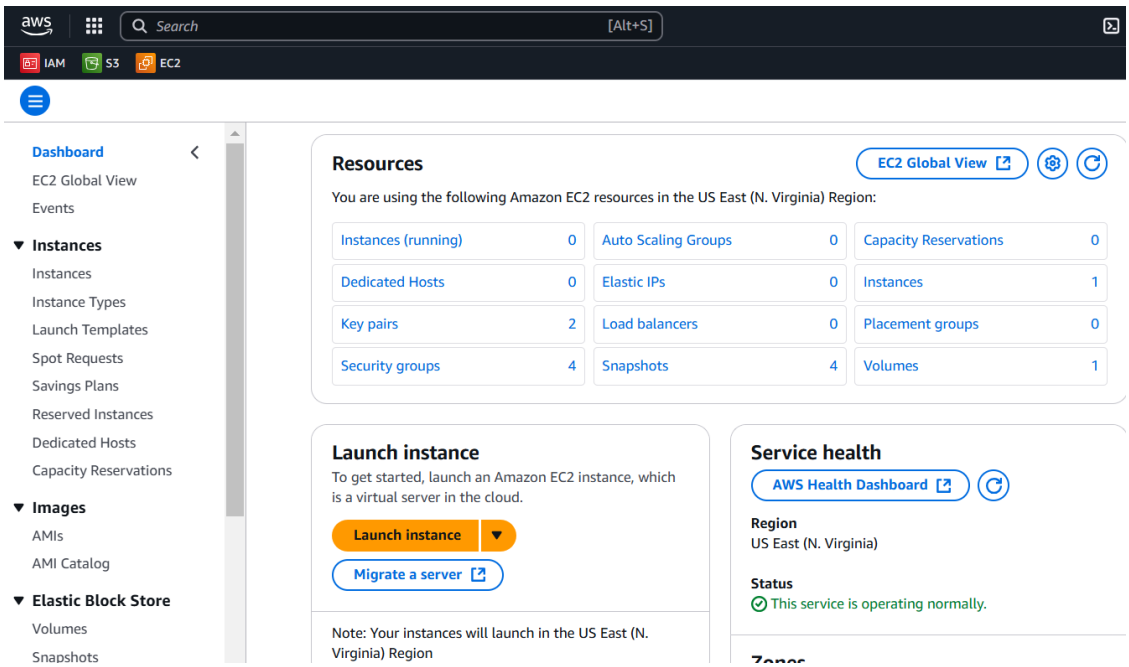

Figure 1: EC2 Dashboard to launch instance.

2. Launch an Ec2 instance with the following

a. Instance Type: t3.2xlarge

b. Operating System: Windows_Server-2022-English-Full-Base

c. Attached Storage: Elastic Block Store (EBS)

3. Create a new key pair with the following options as shown in the figure and save the private key to establish the connection to your instance once the instance is created.

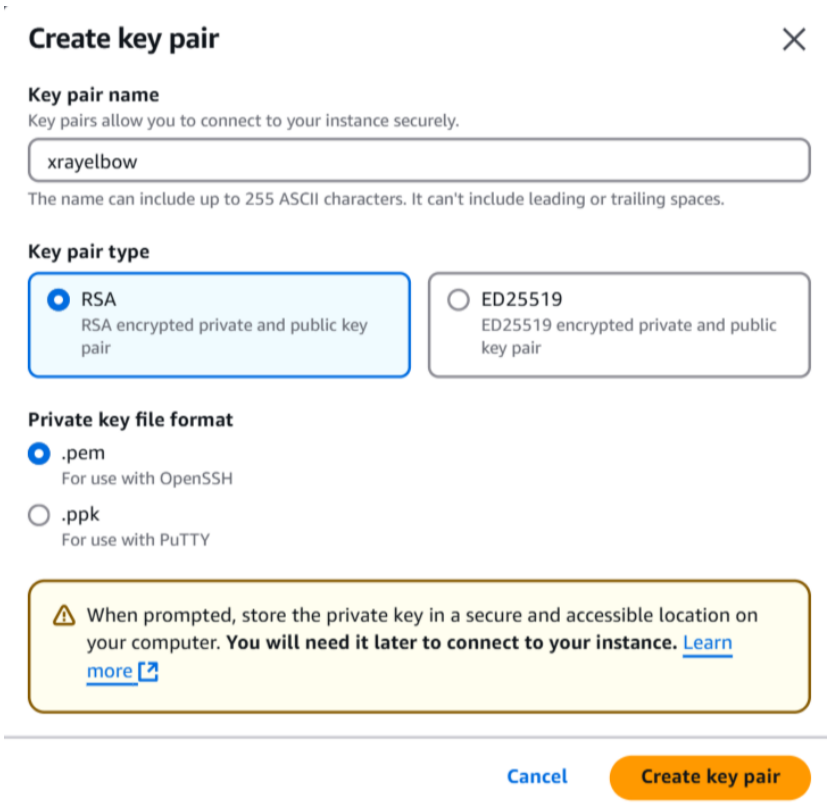

Figure 2: Key pair creation options.

4. Once all this is completed click on launch instance for the instance to be created.

5. Configure Monitoring:

a. Enable AWS CloudWatch.

b. Create custom alarms for CPU utilization >90%.

6. Next install the required software in EC2 instance to train our model

- **Python Setup**:
  - Python 3.9 was installed on AWS instance using the following terminal command:
  - py --version
  - The necessary libraries were installed using pip:
  - py -m pip install tensorflow opencv-python scikit-learn matplotlib seaborn
- **TensorFlow Setup**:
  - py -m pip install tensorflow==2.12.0

- To verify TensorFlow installation, the following code was run in the Python shell: print("TensorFlow version:", tf.__version__)

3.2 Azure VM Setup

1. First create an Azure account and sign in to the Azure Console and go to the VM Dashboard to create a VM click on create a new VM option as shown in the figure.

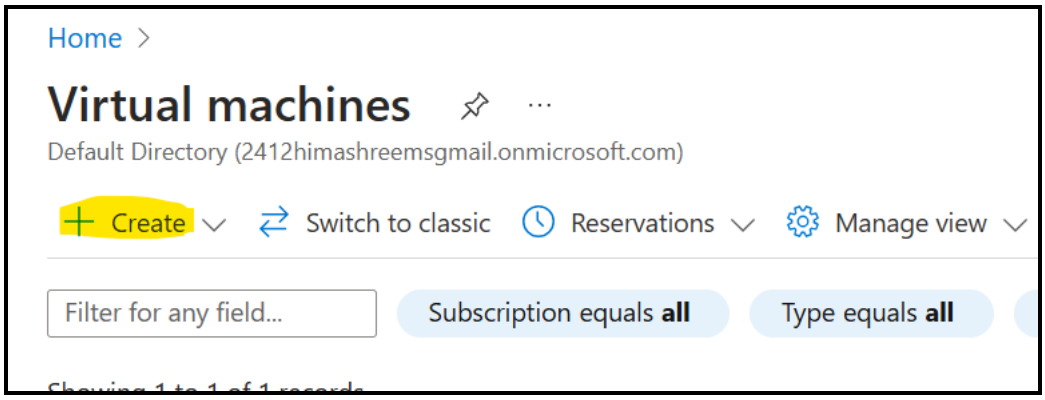

Figure3: VM Console to create new VM

2. Create a Standard_D8_v3 Virtual Machine:
    a. vCPUs: 8
    b. Memory: 32 GB
    c. Operating System: 2022-datacenter-azure-edition
3. Attach Premium SSD LRS for data storage.

When creating the VM choose the following options s shown in figure and create a new resource group as well Once this is done create the username and password to connect to the vm as well in the same console.

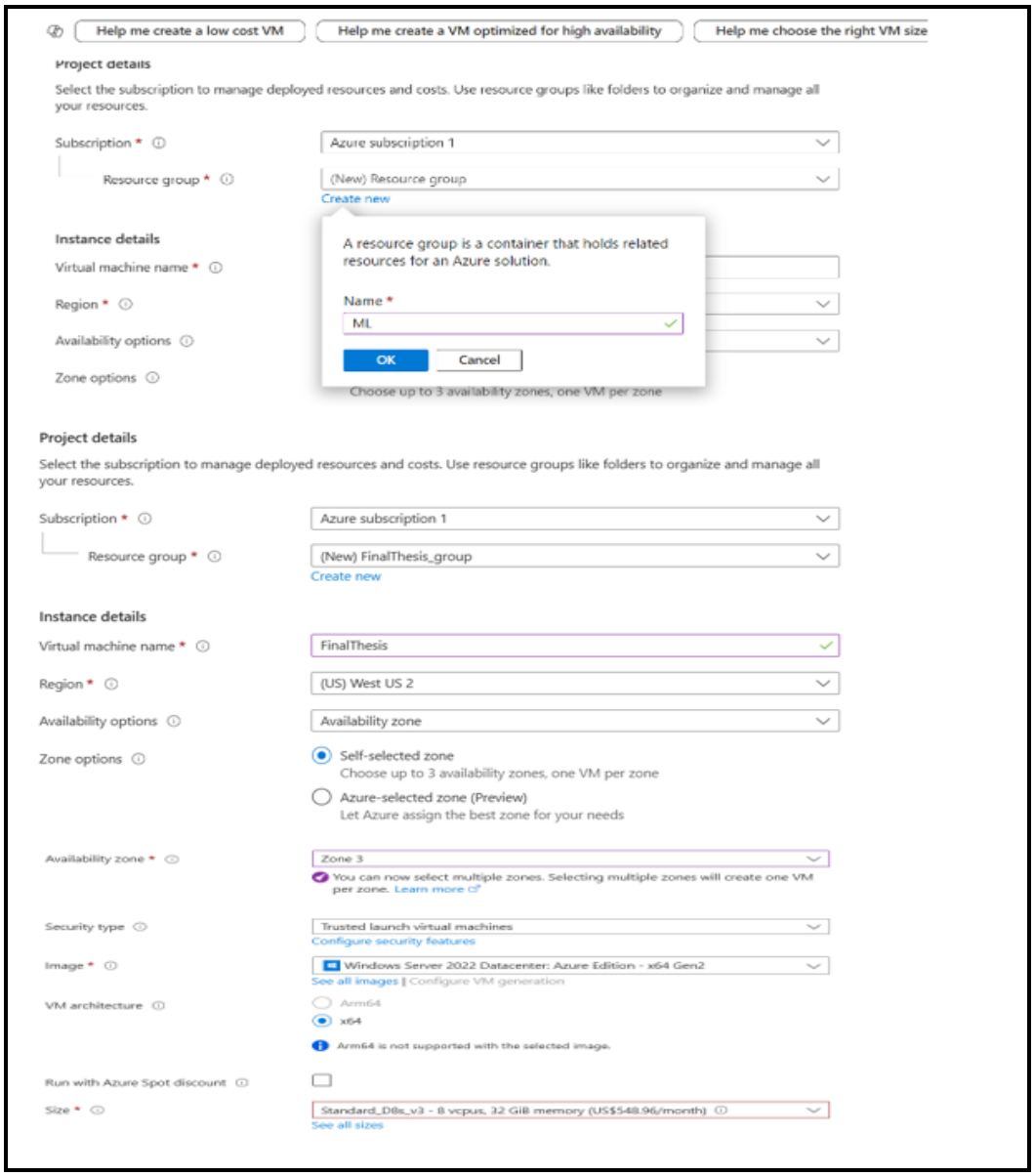

Figure 4: New VM creation options

4. Install Software:
    a. Python 3.9 and TensorFlow using pip through “pip install tensorflow” command
5. Configure Monitoring:
    a. Use Azure Monitor for tracking metrics.
    b. Set up alerts for high CPU utilization.

4. Model Training

1. The dataset for this project is uploaded in the following link http://surl.li/mfafyp You can download this and keep it in your local.

2. Once the required software is installed in both Azure and AWS Copy the dataset from your local and paste it in the directories of the respective platforms.
3. Once all the dependencies are all installed Run the model.py python file to train the model The code created is based on the googlenet algorithm.

5. Monitoring During Training

1. Create own Custom Dashboards in both AWS CloudWatch and Azure Monitor
2. Use AWS CloudWatch to track CPU utilization, memory usage, and disk read write operations.
3. Use Azure Monitor to log similar metrics.
4. Configure alerts for resource thresholds (e.g., CPU >90%).

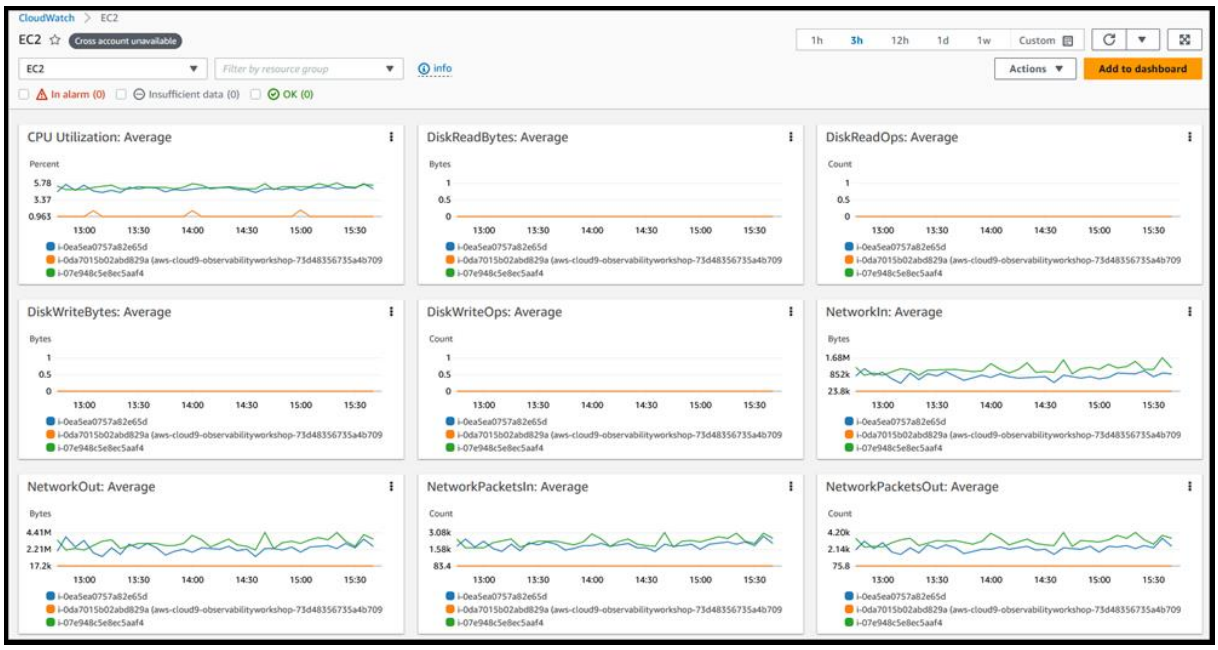

Figure 5: AWS custom monitoring dashboard

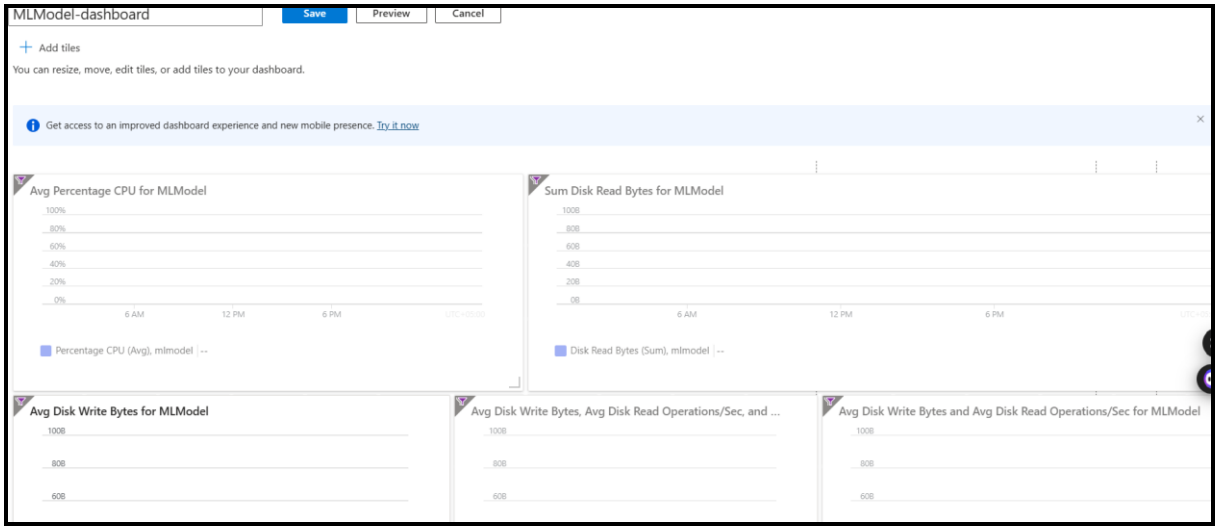

Figure 6: Azure Custom Monitoring dashboard

Once the dashboards are created, we can get the data from training the model which is used in this project to evaluate the performance of both the Cloud platforms and compare.