National College of Ireland

Configuration Manual

MSc Research Project
MSc in Cloud Computing

Pushkar Rahane
Student ID: X21177279

School of Computing
National College of Ireland

Supervisor: Dr. Aqeel Kazmi

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Pushkar Rahane
Student ID:	X21177279
Programme:	MSc in Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Dr. Aqeel Kazmi
Submission Due Date:	18/09/2023
Project Title:	Enhancing Cloud Efficiency using Intelligent Autoscaling Algorithms
Word Count:	650
Page Count:	3

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:	Pushkar Rahane
Date:	18th September 2023

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

Pushkar Rahane
x21177279

1 Objective

The Long Short-Term Memory (LSTM) algorithm came out on top in our study across all criteria. Its capacity to identify complex patterns, reduce lags, mistakes, and waste flops demonstrated its flexibility and efficiency in the auto-scaling scenario. This assessment emphasises the need of choosing an effective algorithm for auto-scaling, with LSTM proving to be a trustworthy option for responding quickly and minimising errors, delays, and resource waste flops in the context of workload prediction and resource provisioning.

2 Tools and Technologies Required:

Python is the programming language used for the project because of its adaptability and extensive library ecosystem. Python can be downloaded and installed from the official website at (https://www.python.org/downloads/).

Importantly, the system runs without issue on Windows, macOS, and Linux (Ubuntu 22.04 or later is preferred). It is not dependent on the operating system. Additionally, the project can be hosted and carried out on AWS (Amazon Web Services) EC2 utilizing Cloud9, improving accessibility and cooperation.

2.1 Required Packages:

Figure 1: Packages

The system's structural support is provided by a significant group of software elements that collectively make up the auto-scaling project. These components include TensorFlow for deep learning models, scikit-learn for traditional models, NumPy for data manipulation, seaborn for data visualization, and HTTP Requests for seamless communication. The project uses Python as its flexible programming language, which supports strong development.

2.2 Code Editor or Integrated Development Environment (IDE):

- VS Code
- Jupyter Notebook
- AWS Cloud9

3 Implementation

To successfully implement the auto-scaling mechanism, follow these steps:
Install Dependencies: Execute the following command to install required Python libraries

pip install -r requirements·txt
or
pip3 install -U tensorflow scikit-learn numpy matplotlib flask requests seaborn

Run the Program (Algorithm-specific Process): Navigate to the project directory using the terminal and execute the following commands:

For Parallel Processor:- python parallel_processor.py
For Serial Processor:- python serial_processor.py
For Server:- python server.py
For Client:- python client.py

Plot the Graph: After running the necessary components for each algorithm, generate graphs to visualize the results. Execute the following command:

python graph.py

These instructions will enable you to successfully install the necessary dependencies, execute the necessary code for each algorithm, and provide useful graphs to illustrate the operation of the auto-scaling mechanism.

4 Evaluation

We conducted a thorough analysis of various algorithms in the context of the auto-scaling mechanism to evaluate their performance in terms of important metrics including response time, total delay, total error, and total wasted flops. Three algorithms were compared: Long Short-Term Memory (LSTM), Random Forest (RF), and Decision Tree (DT).

Response time is the time it takes for a system to manage a workload and respond to a client, affecting user satisfaction. Total delay accounts for cumulative time lag. A lower error rate indicates accurate resource allocation and prediction, promoting dependability. Reducing Total Floating-Point Operations (FLOPs) waste improves resource usage, supporting cost effectiveness and environmental sustainability.

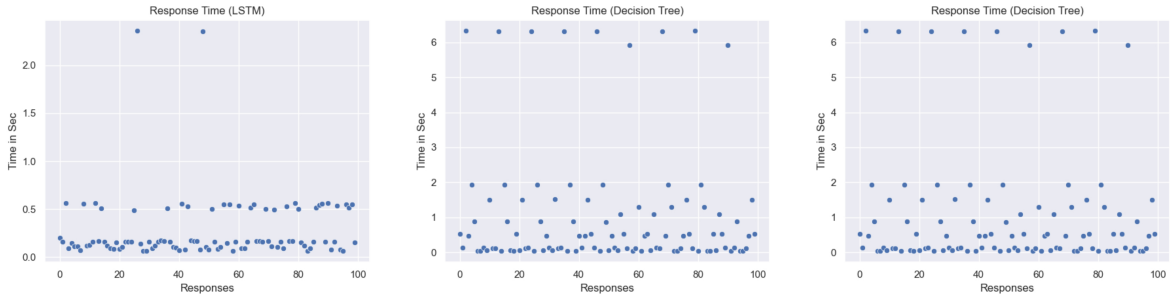

Figure 2:

Figure 3:

5 Results

The study found that LSTM demonstrated exceptional efficiency in response time, allowing quick prediction and resource allocation decisions. RF had moderate response times due to ensemble-based approaches, while DT had slower response times due to oversimplified designs. LSTM had the lowest overall error but struggled with handling complex workload patterns. LSTM excelled in resource allocation, minimizing waste in total waste flops.

References

python Installation

URL: `https://www.python.org/downloads/`

Packages installation using pip

URL: `https://packaging.python.org/en/latest/guides/installing-using-pip-and-virtual-environments/`