

Configuration Manual for running the Ensemble Q-Learning Algorithm using Python based EdgeSimPy Simulator

MSc Research Project
MSc in Cloud Computing

Rakesh Malik
Student ID: 21172692

School of Computing
National College of Ireland

Supervisor: Mr. Rashid Mijumbi

National College of Ireland
Project Submission Sheet
School of Computing



Student Name:	Rakesh Malik
Student ID:	21172692
Programme:	MSc in Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Rashid Mijumbi
Submission Due Date:	20/12/2023
Project Title:	Configuration Manual for running the Ensemble Q-Learning Algorithm using Python based EdgeSimPy Simulator
Word Count:	XXX
Page Count:	2

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:	
Date:	31st January 2024

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):	

Prerequisites

Before running the simulator, ensure you have the following installed:

- Python (version ≥ 3.6)
- Required Python packages: `numpy`, `pandas`, `matplotlib`
- DQN library for reinforcement learning

Setup

1. **Clone the repository:**

```
git clone https://github.com/EdgeSimPy/EdgeSimPy.git
```

2. **Install dependencies:**

```
pip install -r requirements.txt
```

3. **Download the DQN library:**

Download the DQN library from [here](#) and place it in the project directory.

Running the Simulator

Execute the following steps to run the simulator:

1. **Open a terminal and navigate to the project directory.**

2. **Run the simulation:**

```
python simulator.py
```

This will initialize the simulation using the parameters defined in `edge_computing.json` and execute the Q-learning algorithm for task migration.

3. **Monitor the simulation:**

The simulation progress and metrics, including power consumption and Q-learning logs, will be displayed in the terminal.

4. **View Results:**

The simulation results will be saved as a CSV file named `QL_logs_replybuffer.csv`.

Additionally, power consumption graphs will be saved as `Qlearning_migration_power_consumption`.

Simulation Parameters

- **edge_computing.json:** This JSON file contains the simulation parameters, such as edge server specifications, service details and Q-learning hyperparameters.

Code Structure

- **DQN.py:** This file contains the implementation of the DQN library used for reinforcement learning.
- **simulator.py:** The main script that orchestrates the edge computing simulation and Q-learning task migration.
- **ReplayBuffer:** A class for storing and sampling experiences for training the Q-learning agents.

Additional Notes

- The Q-learning algorithm Gao et al. (2020) employs an ensemble approach with the Replay and Buffer mechanisms for decision-making.
- Ensure you have the required permissions to execute scripts in your environment.
- Customize simulation parameters in `edge_computing.json` for different scenarios.
- Experiment with Q-learning hyperparameters Bardenet et al. (2013) in `EnsembleDQNAgent` to observe their impact on the simulation.
- Feel free to customize simulation parameters, explore the code, or refer to the documentation for further insights into the implementation.

References

- Bardenet, R., Brendel, M., Kégl, B. and Sebag, M. (2013). Collaborative hyperparameter tuning, *International conference on machine learning*, PMLR, pp. 199–207.
- Gao, Z., Hao, W., Han, Z. and Yang, S. (2020). Q-learning-based task offloading and resources optimization for a collaborative computing system, *IEEE Access* **8**: 149011–149024.