National College of Ireland

Using Machine Learning in Edge Computing for Optimizing Resource Scheduling

MSc Research Project
MSc Cloud Computing

Bhavna Jasmine Maria Rathna Kumar

Student ID: 22185101

School of Computing
National College of Ireland

Supervisor: Rashid Mijumbi

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Bhavna Jasmine Maria Rathna Kumar
Student ID:	22185101
Programme:	Masters In Cloud Computing **Year:** 2023
Module:	MSc Research Project
Supervisor:	Rashid Mijumbi
Submission Due Date:	16-09-2024
Project Title:	Using Machine Learning in Edge Computing for Optimizing Resource Scheduling
Word Count:	7257 **Page Count** 20

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:	Bhavna Jasmine Maria Rathna Kumar
Date:	16-09-2024

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):	

Using Machine Learning in Edge Computing for Optimizing Resource Scheduling

Bhavna Jasmine Maria Rathna Kumar

22185101

Abstract

Edge computing is an emerging trend that has faced challenges in the allocation of resources in light of its dynamic and distributed nature with implications on latency, computational power and energy. Central data processing is no longer sufficient for high-real time applications such as IoT, self-driving cars, and smart cities. To address these problems, this project proposes a framework that applies machine learning approaches such as Random Forest, Decision Tree, AdaBoost, and Gradient Boost to improve resource scheduling. The purpose of the project is to establish whether the integration of fusion models and cloud computing can enhance resource management. This project established the most effective fusion model through experimentation and implemented it on AWS to prove that cloud-based systems enhance computation and dependability. This framework directly solves the problem stated in the research question of how to improve the resource scheduling for enhancing edge computing services and how the advanced analytics and cloud infrastructure can be used to improve the performance in several latency critical applications.

1 Introduction

1.1 Overview on Edge Computing

Edge computing is a new concept in data processing wherein instead of transmitting data to central data centers for processing, computations are done near the source of data. In contrast, edge computing models do not transfer data to a far-off data center for computation, as seen in normal cloud computing. This approach is very useful for applications that are latency-sensitive and have real-time processing needs such as IoT devices and autonomous vehicles and smart cities (Xiong, et al, 2020). Edge computing refers to the process of data creation at the network's perimeter by devices such as sensors, smartphones, IoT devices, and self-governing vehicles. This data is then transferred to the edge nodes which are the intermediate processing elements like gateways, routers or edge servers which are specialized in analysis of the data. The smaller edge data centers located at the edge as compared to the centralized data centers offer extra computing power and storage. These components are connected through LAN, WAN, and the internet to enable efficient transfer and analysis of data at the network's periphery (Deng, et al, 2020).

1.2 Need for Optimizing Resource Scheduling for Edge Computing

Resource scheduling in edge computing is vital in order to maximize the its benefits in the enhancement of computational capabilities as well as low latency. By placing more computing resources near the data source, edge computing reduces the load on central data centers and shortens the time required to deliver services. Resource scheduling is a dynamic element that makes computation jobs tractable by implementing scheduling procedures available or in demand to gain maximum tractability (Wang, et al, 2021). It also supports the optimization of energy usage and energy consumption in edge devices, which is important for counts as longer battery life, critical for sustainability of the Internet of Things (IoT) setup. In addition, the schedules are thereby optimized to promote scalability: it becomes easy to incorporate new devices and application into the system without straining the framework. Due to a smaller number of interconnections and better resource utilization, the edge computing can offer more reliable and faster services which are mostly required in applications like autonomous cars, smart cities, and industries (Mao, et al, 2020).

1.3 Need for resource scheduling in edge computing

This concept of edge computing is informed by an increased explosion of data generation at the frontier of the chosen network in the resource scheduling. Typical centralized cloud computing environments are limited by latency and bandwidth when handling the vast data output of IoT devices, sensors, and other edge devices. Edge computing solves these limitations by distributing computing and storage and moving computations closer to where the data is produced. This close location minimizes the time and resources needed for data processing and decision-making since these can be done in real time or close to it (Qian, et al, 2020). Edge computing uses an efficient strategy of sharing computing resources across edge nodes to manage and response to different workloads and application requests. This approach does not only increase the system performance and stability but also increases the scalability and yet low cost since it does not require a lot of data transfer and processing in a central way. Therefore, it can be concluded that edge computing is critical to efficient resource management as well as the adaption of computing architectures based on the location of data sources (Teoh, et al, 2021).

1.4 Problem Statement

Edge computing is a new model that has been adopted in the current world of data processing where data is processed near the source rather than in central data centers. This approach is crucial for low latency applications that involve real time processing like IoT devices, self-driving cars and smart cities. However, the problem of resource management and its optimization at the edge of the network is still an open issue. The standard techniques used in managing resources are inadequate in dealing with the dynamic and distributed characteristics of the edge environments and this results in suboptimal utilization of computational resources and energy. Current approaches have limitations in terms of latency, computational costs, and energy consumption, which become a problem when dealing with large amounts of data from IoT devices, self-driving cars, and smart cities. To overcome these problems, it is necessary to develop the most efficient resource schedule which can be

implemented with the help of machine learning techniques. This project seeks to address the existing gap in the management of Edge Computing Resources by presenting a comprehensive model that encapsulates different machine learning techniques such as hybrid models, and implement them in a cloud environment. In terms of scalability, the AWS is used to improve the efficiency of the resource scheduling in edge computing systems while RestAPI (Bottle API) is used to develop real time predictions.

1.5 Aim

This research project aims to develop a framework to enhance the management of resources in edge computing through the incorporation of various machine learning strategies that are the fusion models and the best model of is deployed in cloud environment. The proposed project is to address the research void that exists in the optimization techniques employed in edge computing. In this manner, this model will assist in the development of better and more efficient edge computing structures suitable for several applications.

1.6 Research question

Do fusion models integrate with cloud helps Optimizing Resource Scheduling for better Edge computing services?

Fusion models are combinations of the multiple models of machine learning algorithms that are implemented by combining two or three algorithms to achieve the best performance in terms of accuracy.

1.7 Objectives

- To conduct a detailed literature survey based on edge computing, resource allocation in cloud environment.
- To implement machine learning algorithms like, Random Forest, Decision Tree, Adaboost, and Gradient Boost, construct fusion models of the above-declared algorithms and evaluate.
- To deploy the best fusion model saved in JSON format in AWS cloud environment for optimizing resource allocation process.
- To utilize Flask API and finding the best server for the allocated resources in the AWS cloud environment.

1.8 Importance of Research

If the full potential of edge computing is to be deployed, then there details an efficient resource allocation method . Organisations can consequently adjust their operating costs and elevate edge application availability, robustness and scalability if their resource allocation procedures are optimised. With these improvements the capability of scheduling resources to support latency sensitive applications in the different edge computing paradigm is now possible to include; autonomous systems, augmented-reality and real time analytics.

2 Related work

2.1 Emerging Trends and Innovations in edge computing

The future of the decentralized computing architecture is defined by the trends and innovations in the edge computing. Among the trends it is possible to identify the application of 5G technology which will greatly contribute to the development of edge computing due to the possibility of using ultra-low latency and high bandwidth connections. This paves way for real time data processing and applications in various sectors such as automotive, manufacturing and others such as self-driving cars, industrial automation (Talaat, 2022). Also, the integration of artificial intelligence and machine learning in edge computing is enhancing decision making as well as predictive analysis at the network's edge. These technologies help in better utilization of resources, dynamic workload management, and adaptive systems which can work based on the changes in environment. New architectures are being proposed to enhance the edge-cloud collaboration, thus, facilitating the interconnection between the edge devices and the cloud computing core. All these trends in the future propel innovation and improvement of edge computing systems and increase the possibilities of even more effective, adaptable, and secure edge computing applications (Tan, et al, 2021).

2.1.1 Recent Technologies used for Resource allocation in edge computing

In other studied instances of automotive edge computing, Gao, J. et al. , (2022), offer a two-layer solution for offloading scheduling and resource management. The issue with their project can be pinpointed at the general issues related to resource management in vehicular environments where the vehicles themselves are considered as edge nodes with limited processing capabilities. The proposed method attempts to improve the effectiveness and dependability of automotive edge computing systems through the integration of offloading scheduling with resource allocation. Their work uses the GD method as the proposed technique, which attempts to minimize the latency with energy consumption of VEC system for different network parameters aiming at improving the effectiveness of resource scheduling. Nevertheless, it will be useful to compare the suggested algorithm with other innovative approaches not only to task offloading with resource allocation in VECs but to analyze the ways to adjust the weight factor of the objective function to find the best trade-off between delay and energy consumption since their research also compares their approach with several existing benchmark systems.

There is a paper of Aghapour, Z. et al. , (2023) which is one significant contribution in this field; the paper focuses on presenting an algorithm for resource allocation and task offloading for distributed AI execution tasks in IoT edge computing environment through deep reinforcement learning. An analysis of how the suggested approach in this work performs on several IoT device deep learning workloads is done with different sophisticated cloudlet server's capabilities. In their work, the authors identify that simulation results revealed that

the proposed algorithm yields enhancement in criterion than techniques such as full local, full offload and Jointly Resource allocation and computation Offloading PSO (JROPSO) with percentage improvement of 92 %, 17 % and 12 % respectively. It also has the least delay and energy utilization. However, the project states that regardless of the variability of the offloading choice, it belongs to the NP-hard MINLP (Mixed Integer Nonlinear Programming). Therefore, the matter demands correct algorithms to address it.

Like this, Alfakih, T. et al. , (2020) proposed a deep reinforcement learning based on SARSA to provide a framework for the resource allocation & job offloading for the Mobile Edge Computing. Their work is about the enhancement of the resource management decisions for the betterment of MEC systems especially in the conditions with dynamic features of the network as well as variable resources with the help of reinforcement learning methods. This is mainly works on the application of reinforcement learning, moreso, the OD-SARSA for job offloading and resource allocation for mobile edge computing (MEC). However, it work highlights that MEC entails a suitable compute offloading that is because of the variability and unpredictability of the mobile edge networks that may lead to high energy consumption and challenges with latency demands.

In Wang, T. et al, (2022) a reinforcement learning based optimization for the scheduling of mobile edge computing is proposed. This project's approach incorporates performance and scalability of optimization in resource allocation in mobile edge environments. Their project focuses on the problem of scalability in edge computing systems and applies reinforcement learning approaches to solve it.

However, the other long term optimization based on the RL for the schedule is given by Xiong, X. et al (2020) where the major objective being to minimize the long term average of the needed resources and the time taken to complete the jobs in the IoT edge computing system. All of them have involved the computing resource allocation, job queue status, adjusting indication, requested computing resources, and backlog of open jobs in the state information and the authors have claimed that their proposed approach is effective in reducing the complete time of the jobs and the consumption of the resources in the IoT edge computing system compared to other referred policies. However, their project underscores that it regarding the resource allocation strategy it suggested that they endeavour to employ a strategy that arrives at minimizing the mean of resource usage and the mean time for tasking overtime in the long run this means that this necessitates research on the specific resource allocation for IOT edge computing systems.

2.2 Research using Machine Learning Models in Resource Scheduling

Djigal, et al, (2022) project provides an extensive examination of machine learning and deep learning-based resource allocation algorithms in mobile edge computing (MEC). Initially, the authors provide lessons that showcase the benefits of using ML and DL in Mobile Edge Computing (MEC). Then, the authors proceed to describe the technologies that facilitate effective machine learning and deep learning training and inference operations in MEC environments. Then, authors present a detailed review of the most recent works that applied ML and DL techniques for the purpose of resource management in MEC. The analysis is

grouped into three main sections based on the ML/DL methods used in offloading, scheduling, and joint resource management. Finally, the authors meet the research challenges and opportunities that can be suggested for the use of ML and DL to allocate resources in mobile edge computing networks.

Wang, et al, (2020) research aims at enhancing the task and resource management in MEC networks via the use of machine learning. This process includes DRL, DQN, and CNN for the proper allocation and optimization of resources and tasks. Therefore, the dynamic and diverse nature of the MEC networks' characteristics is addressed in regard to managing the utilization of the resources and minimizing latency. It was also established that the costs are lower, the latency time is short, and there is better resource management as compared to the conventional methods. These techniques enhance the ability of the system to adapt to the changes in the network and hence make the system stable in its performance. Following are the conclusions derived from this research concerning the application of DRL and DQN for MEC systems to enhance their efficiency in their dynamic environments. Hence, such algorithms show that the network management of this type can become more effective in the future of MEC systems' evolution. The project describes how machine learning allows for the evolution of the schedule of resources in edge computing for the development of more efficient applications across industries and real time analytics, automation and augmentation. So, employing the mentioned techniques, MEC systems will be able to serve the highly latency-sensitive applications more effectively and hence enhance the efficiency and practicality of the systems.

The article by Hussain, et al, (2020) is a systematic and elaborate review of the ML and DL resource management techniques used in cellular wireless and IoT networks. The first focus is on the challenges which are related to the resource management in the cellular IoT and low-power IoT networks. The authors then review the traditional approaches employed in IoT networks for resource management and explain why it is possible to apply ML and DL in such management. Subsequently, the authors discuss in detail the current state of affairs in terms of the ML- and DL-based approaches to controlling the resources in wireless IoT networks. Also, the authors present solutions that are appropriate for HetNets, MIMO, and D2D communications as well as NOMA networks. Moreover, the authors also present the possible research directions in applying the ML and DL for the management and allocation of resources in IoT networks.

2.3 Research gap and novelty

Previous works have analyzed numerous machine learning models and their use in resource management. There is limited literature that presents an overall framework of these models in real-world edge computing systems, especially with the use of fusion models. For example, Gao et al. (2022) have utilized only one machine learning algorithm for vehicular edge computing while Aghapour et al. (2023) have used single machine learning algorithm for IoT scenario while there is possibility to get better results by integrating several algorithms. However, work by Alfakih et al. (2020) and Wang et al. (2022) has shown the effectiveness of deep learning and reinforcement learning in resource management but little is known of how these can be practically applied in cloud platforms such as AWS. The application of

fusion models in the cloud for the management of resource scheduling in edge computing is relatively less explored (Djigal et al., 2022). This gap is important because the deployment of cloud can greatly improve the scalability, flexibility, and real-time processing of edge computing systems. Hence, this research aims at addressing these gaps by proposing a framework that involves the integration of different machine learning techniques, the fusion models, and the implementation of the best models on the AWS cloud environment. This work intends to show how the combined methods are beneficial to enhance the management of resources, scalability, and real-time processing for edge computing applications as a way of filling the gap that exist in optimization methods.

3 Research Methodology

The first aspect of the investigation is on the usage of the work which refers to the method and the tool used in tackling research question. This section explains the general approach used in the resource scheduling optimization conducted in the context of edge computing. According to the works' objective the project aims to develop context-sensitive self-organizing algorithms with resource allocation based on network conditions and data source and workload characteristics.

3.1 System design

AWS is chosen for its scalability, reliability, and cost-effectiveness, with Flask for its flexible architecture, and machine learning techniques like Random Forest, Decision Tree, AdaBoost, and GBM for their strengths in handling diverse data types. The Kaggle's dataset on 5G Quality of Service contains various differentiator concerning the quality of the 5G network. The main file "Quality of Service 5G.csv" has several fields/parameters including Latency, Bandwidth, Signal Strength, Packet Loss and User Equipment parameters. They are critical to identify the quality of service in 5G networks, and to comprehension network performance in various scenario, as well as resources management. It is useful for research in the field of wireless mobile networks and specially in the construction of the 5G network. The first main experiment will be focused on identifying the best models for the resource scheduling. Totally four machine learning algorithms and one neural network algorithm and one fusion based algorithm will be implemented for this research for resource scheduling. Machine learning and deep learning algorithm will be implemented using scikit learn package where all the methods will be called from the package. The fusion model will be created using voting classifier which will be implemented using the scikit learn package. The voting classifier will create a single best model (fusion) learning from several models based on voting majority to predict resource scheduling efficiently and precisely. Theoretically this approach will reduce bias and variation and will increase the performance. This increase in performance will be by combining the predictions of many models. The fusion model will forecast the resource scheduling based on the voting majority. We will use hard and soft voting in our fusion model. The fusion model created for the resource scheduling is deployed in AWS cloud environment using Flask application. FastAPI is employed for the development of APIs which can enable interoperability and obtain the model. The integration of AWS and Fast API used in the project helps to provide high-quality, efficient, and scalable

solutions in the field of machine learning to meet the performance and operational objectives in the changing conditions. Fusion models combining these algorithms will be evaluated based on accuracy, precision, recall, F1 score, and time, with the best model deployed on AWS using EC2 instances and RestAPI (Bottle API) for real-time predictions and secure data management. This approach aims to enhance resource scheduling efficiency in edge computing environments.

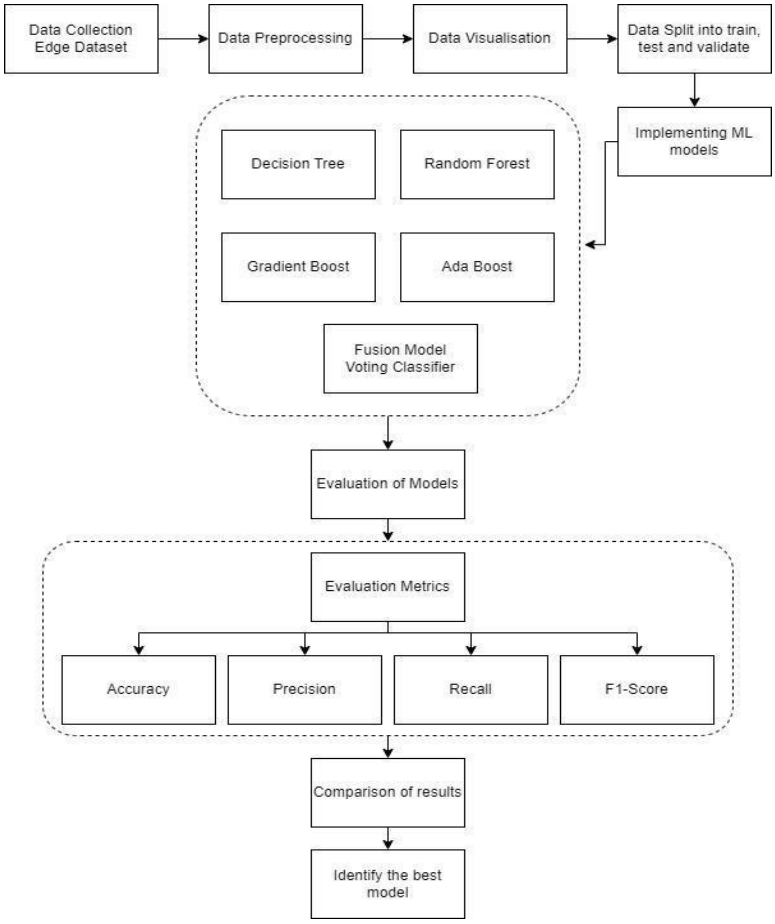


Figure 1 Implementing ML and Combined models- Step 1 of the Implementation

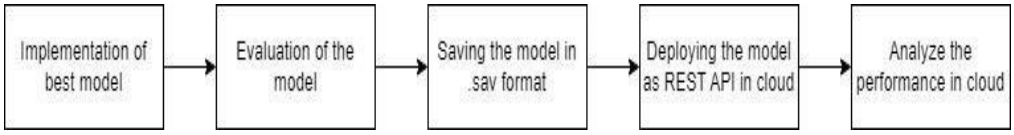


Figure 2 Implementing ML Model in cloud- Step 2 of the Implementation

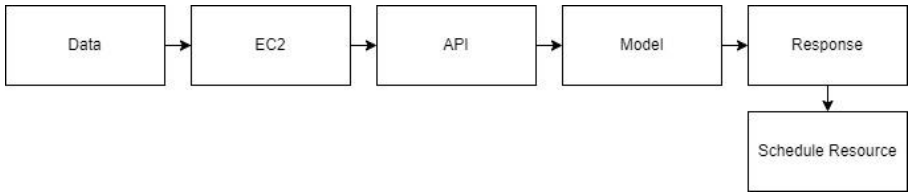


Figure 3 Implementing and Integrating API with the Model- Step 3 of the Implementation

4 Design specification

4.1 Choosing the model for the deployment

The selection of the machine learning model is crucial for optimizing resource allocation in edge computing. To address this, the project evaluates various models, including Random Forest, Decision Tree, AdaBoost, and GBM, each known for its distinct strengths in handling different types of data and problem-solving scenarios. By developing and testing fusion models that combine these algorithms, the project aims to identify the most effective approach for accurate, efficient, and scalable resource management. The chosen model will be deployed in the AWS cloud environment, leveraging its scalability and real-time capabilities to enhance edge computing performance.

4.2 Implementation of fusion model in cloud

4.2.1 AWS Cloud Service

AWS is one of the most popular and diverse cloud computing services which provides a multitude of IT solutions. AWS has gained its reputation due to the pay as you go model which makes the users pay for what they actually use without making huge investments in infrastructure (Chakraborty et al., 2023). It has a wide range of data centers which allows users to launch applications and services closer to the consumers thus decreasing the response time and increasing the level of user satisfaction. Some of the services offered by AWS are computation, storage, database, analytics, machine learning and networking among others to meet different business needs (Singh et al., 2023). It is also secure as it provides data encryption, compliance certifications, and several security tools in order to protect the data and abide by the set legal standards. To that end, AWS is a reliable, large, and dynamic solution for digital transformation because of its constant change and frequent updates so that enterprises can utilize the most up-to-date technologies without having to worry about the management of their IT infrastructure.

4.2.2 Role of RestAPI (Bottle API)

RestAPI (Bottle API) is a fast (high-performance), web framework for building APIs in Python that is simple to learn and use. RestAPI (Bottle API) is known to be simple and effective and shines when it comes to dealing with asynchronous operations which are

essential in the cloud where there is a need to manage multiple requests at the same time (Voron, 2023). It enables the creation and management of good APIs that can easily integrate with the cloud services and applications. The simplicity of using RestAPI (Bottle API) for the API development along with the dynamic nature of Python and its support from the vast library makes it perfect for cloud native application. These include auto generated API documentation where operations are done concurrently, compatibility, speed and microservices (Surendar et al., 2024). RestAPI (Bottle API) helps to overcome the problems of contemporary web applications by creating performant, powerful and documented APIs for the organization's needs, especially since more and more companies are using cloud services.

4.2.3 EC2 Instance

Amazon EC2 is a web service that provides resizable virtual machines in the cloud to organizations and developers to use as per their requirement (Linder et al., 2024). The payment on EC2 is also made in the form of instances used and this eliminates costs that would have been incurred on hardware. EC2 instance types that are available in AWS include the following: Compute optimized, Memory optimized, Storage optimized, and GPU instances which can be used depending on the application's requirements with regard to the processing power, memory, and storage. Some of the parameters of the EC2 instances include multiple operating systems to choose from together with highly configurable security, network, and storage settings. Some of the important characteristics include flexibility, the ability to modify resources as per the need of the hour, and scalability, which means that it is possible to add or remove resources without affecting the application's performance. AWS has made it possible to have been tributed global network that makes it easy to have low latency and high availability in all the regions thus improving the effectiveness of the EC2 instances (Dancheva et al., 2024).

4.2.4 JSON Format

This project identified that JSON is a critical format for deploying machine learning models in AWS cloud environments, especially regarding the portability and to exchange data through APIs. JSON enhances the compatibility with several systems and languages due to effective serialization and deserialization of the ML models at the time of deployment (Hofmeier et al., 2023). This approach saves the model's attributes and configurations, which allows for easy replication of environments in the cloud. JSON also includes meta-data information and all the configurations in the model which enable it to run with high efficiency and scalability. It synchronizes with other AWS services like Lambda functions and EC2 instances as it supports the flexible application design and efficient resource management (Maltsev et al., 2024).

4.2.5 Flask Application

Flask is a python web application framework that is lightweight, easy to learn and yet very powerful. Some of which include URL and HTTP request handling routes, and templating through Jinja 2 which forms the core of building modern and efficient web applications

(Syach et al., 2024). Flask has a simple structure and does not include many built-in instruments, it is possible to expand the functionality of the framework for such tasks as working with a database and creating an API, which makes it possible to use Flask for a variety of tasks. Flask is a WSGI web framework based on Werkzeug which guarantees a proficient request and response handling leading to the efficiency and stability of the framework. Due to its simple architecture and the ability to work with cloud services, Flask can be used for small projects as well as large companies, thus enabling efficient and scalable web application development.

5 Implementation

The experiment chapter focuses on the real-world implementation of the proposed machine learning models for effective resource allocation in edge computing systems. It describes the methods of data cleaning and transformation, model building and optimization, and model assessment with a focus on deploying the top model on AWS and creating the UI using the Flask framework.

5.1 Data Preparation

The dataset was first loaded into the system and the shape of the dataset, data types, and the first few rows of the dataset were checked to understand the structure and content of the dataset. The data is then cleaned to eliminate missing values and duplicates to ensure that the data used was accurate. Some variables were eliminated; for example, Timestamp, Source_IP, and Dest_IP since they are not useful in the analysis and only add up to the dimensionality of the feature space. Categorical variable was transformed to numerical values to feed into the machine learning algorithms through the encode of labels. In addition, the cleaned data was stored to be used in the future in the analysis and modeling of the further stages.

5.1.1 Exploratory Data Analysis

In the course of Exploratory Data Analysis (EDA), the distribution of the target variable ‘Server_ID’ was depicted to check for class imbalance which may affect the performance of the model. To analyze the correlation between the features and to help in the selection of the features and model understanding, a heatmap of the correlation matrix was created. To enhance the performance of the models and manage class imbalance one of the classes in the target variable ‘Server_ID’ was eliminated.

6 Evaluation

The data set was further divided into training, testing, and validation to guarantee the model’s performance was properly analyzed. The training set was employed to build various models while the testing and validation sets provided the evaluation of the models. This was followed by applying feature selection and building and assessing a Decision Tree classifier which was

hyper-tuned using grid search to determine the best hyperparameters. The model's accuracy was measured on the validation and testing sets and the results were presented graphically through confusion matrices. Afterwards, AdaBoost, Gradient Boost, and Random Forest classifiers were employed and fine-tuned in the same manner. All the algorithms are also integrated to create the fusion models and assessed in the same manner as other algorithms. All the models were evaluated on the validation and testing sets to guarantee the effectiveness of the methodology. The following tables give the overall validation and testing results of the algorithms implemented respectively. Table 3 refers to the confusion matrix of Adaboost algorithm with test data and validation data. The following is a bar graph representing the results of algorithms implemented and their validation results.

Table 1 Validation Results

Model	Validation-Time	Precision	Recall	F1-score	Accuracy
DT	0.06	1.00	1.00	1.00	1.00
Adaboost	0.05	1.00	1.00	1.00	1.00
GB	0.08	1.00	1.00	1.00	1.00
RF	0.22	1.00	1.00	1.00	1.00
Fusion Model	0.08	1.00	1.00	1.00	1.00

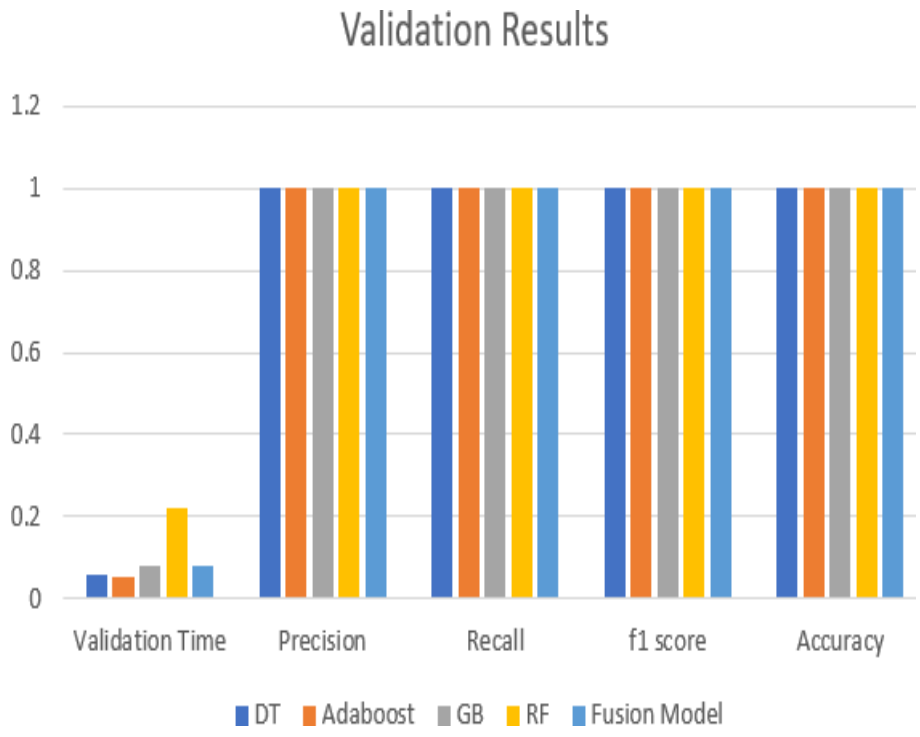


Figure 4 Validation Graph

The table below represents the results of testing of all the algorithms following which is a bar graph of the results.

Table 2 Testing Results

Model	Testing-Time	Precision	Recall	F1-score	Accuracy
DT	0.06	1.00	1.00	1.00	1.00
Adaboost	0.05	1.00	1.00	1.00	1.00
GB	0.08	1.00	1.00	1.00	1.00
RF	0.20	1.00	1.00	1.00	1.00
Fusion Model	0.08	1.00	1.00	1.00	1.00

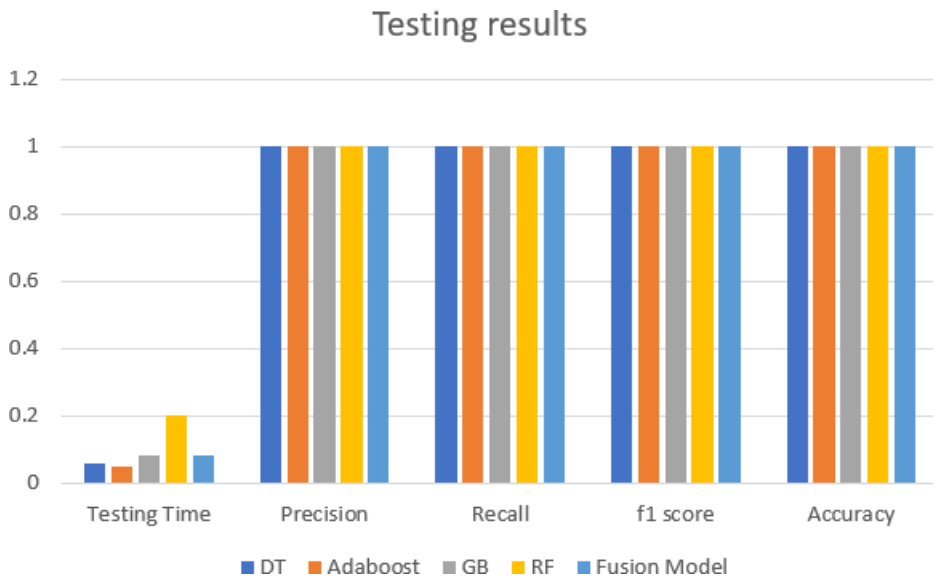
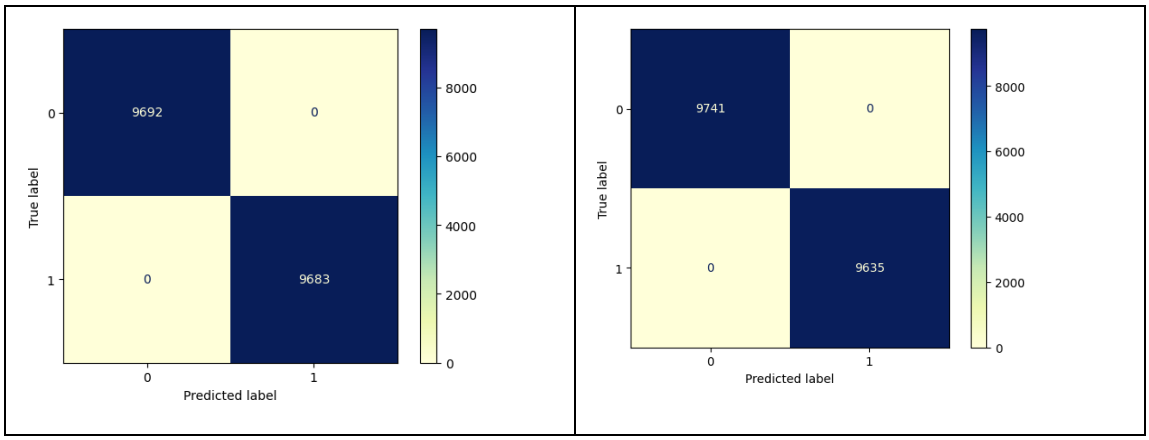


Figure 5 Testing graph

Table 3 Confusion matrix of Adaboost with test and validation data



Confusion Matrix of Test data with AdaBoost	Confusion Matrix of Validation data with AdaBoost

The above figures represents the confusion matrix of Adaboost algorithms with the validation and testing data. All the classifiers such as DT, AdaBoost, GB, RF, and Fusion models achieved a perfect performance with an accuracy of 1.00. on weighed precision, recall, f1 score and accuracy both in the validation and the testing sets. All models had high accuracy, and AdaBoost had the least training (0.30 sec) and testing (0.05 sec) time, hence, making it the most efficient in terms of computational time. Hence, all the models were quite effective, but the AdaBoost was recommended to be the best model considering its efficiency. Therefore, it was selected as the best working model and was saved for possible use. Thus, the proposed method allowed for selecting the most efficient model for resource allocation optimization in edge computing.

6.1.1 Testing the Deployed Service

Once the API was deployed in the EC2 instance, it was tested through the help of Postman in order to check the effectiveness of the API. First, the service was opened by using the given URL and several cases were used to try to get the response from the API and check the accuracy of the predictions. The testing was done by making POST calls to the API endpoint where we checked the response to see if the returned predictions were the same as what was expected. Below figure represents the code snapshot of testing of deployed services.

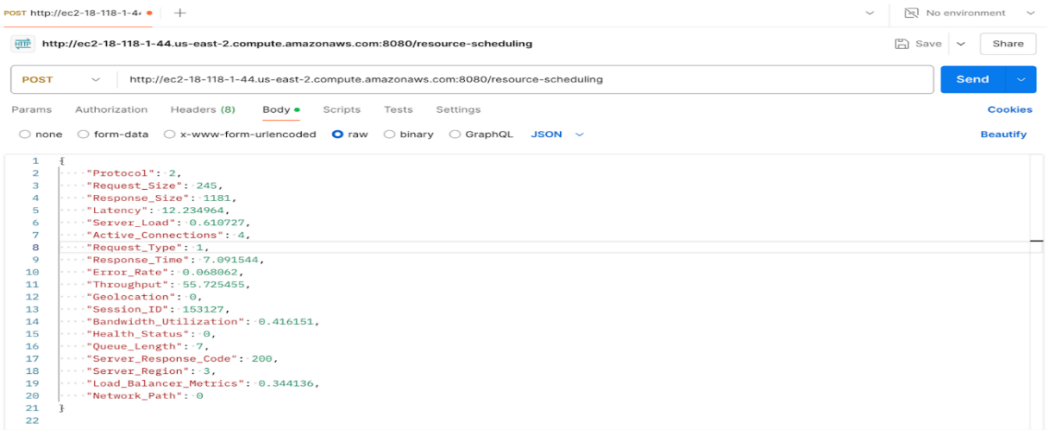


Figure 6 Testing deployed service

The success of the deployment was known as the API returned the correct results in the form of JSON which proved that the Bottle API was properly coupled with the pre-trained model and was operative. The successful deployment and testing of the API illustrate that AWS cloud infrastructure is suitable for hosting and managing any machine learning applications.

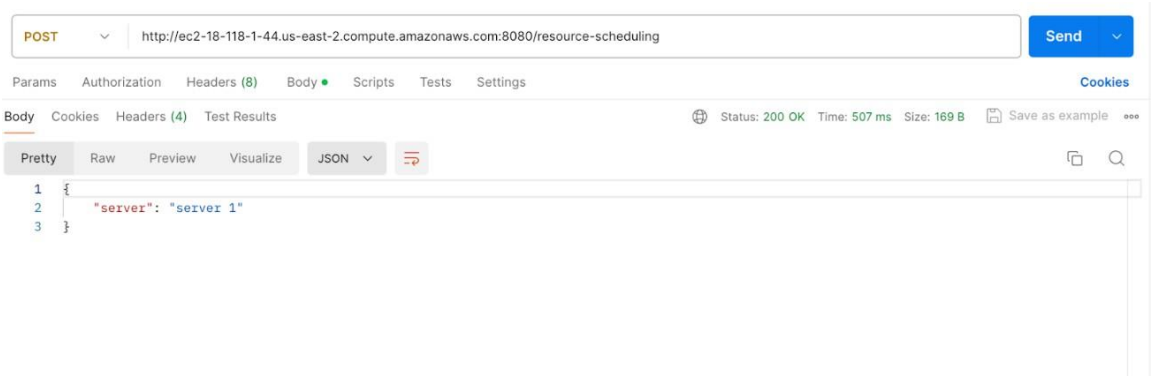


Figure 7 Hosting ML applications

6.1.2 UI Interface and Execution

The Flask application has an implementation of frontend user interface in the form of a web page intended for interaction with a resource scheduling API. The application includes two primary HTML templates: The following is the input. html and output. html. The input. This HTML template displays a form that is built with Bootstrap and enables users to provide different parameters that relate to resource scheduling, like the Protocol, Request Size, and Response Size, among others. Upon submission, the form data is sent to the backend API at http://ec2-18-118-1-44.us-east-2.compute.amazonaws. com:8080/resource-scheduling through a POST request with a JSON value. The output. After that the page also design its output according to an HTML template and shows the server ID and the parameters that are to be submitted and in an orderly manner. This way, the data submission and response display are done efficiently and in an organized manner making the process convenient for the user. The following is the frontend UI where the input parameters are passed through form data.

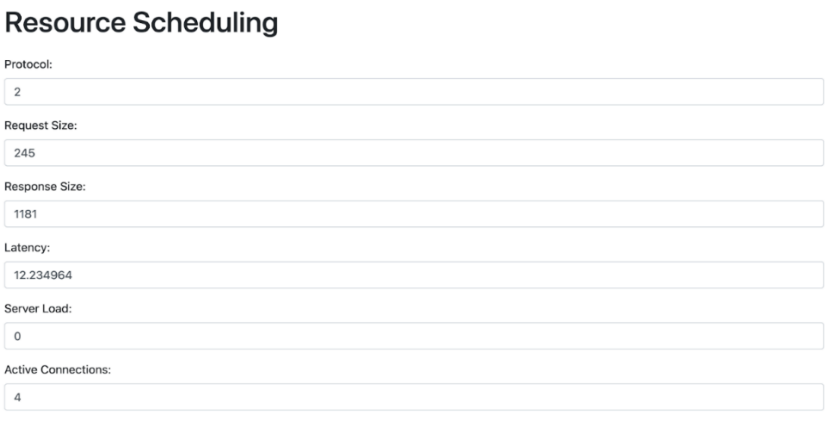


Figure 8 Resource scheduling frontend

The following is the output generated by the service hosted in the AWS EC2 instance provided.

Figure 9 Resource scheduling output

6.2 Summary

Each of the models like Decision Tree, AdaBoost, Gradient Boost, and Random Forest was tested for its effectiveness in resource scheduling for edge computing in the experimentation phase. All models got the highest possible score in the validation and testing steps and the fastest and most effective model is AdaBoost. The chosen AdaBoost model was implemented on the AWS using a Bottle API to perform predictions and was thoroughly tested. The deployment proved that the cloud resources and machine learning played a significant role in improving the edge computing.

7 Critical Analysis

7.1 Analysis of this Cloud Integration

Cloud computing integrated with edge computing is an important new concept for managing and enhancing the use of resources. Using cloud infrastructure and especially AWS, the project targets an essential issue in edge computing, which is the proper management of resources. Through the implementation of the AdaBoost model, the use of AWS EC2 and its capabilities on edge computing is highlighted. This setup also enhances the resource sharing efficiency while at the same time guaranteeing efficient processing and real time data analysis which can be very useful in IoT devices and autonomous systems. The non-intrusive and easy integration and expansion of the cloud infrastructure support the need for edge computing systems.

The utilization of AWS for the implementation of Bottle API points out the significance of cloud computing in application development such as adaptability, cost, and maintainability. AWS offer a reliable infrastructure that assists in the running of machine learning models and aids in the management of resources. Thus, the successful implementation of the API proves that cloud integration can efficiently solve the issues of edge computing and manage distributed resources to improve the overall system performance.

7.2 Analysis of the Integration of the Cloud with Machine Learning and UI Interface

Machine learning models that are combined with cloud infrastructure allow for the improvement of resource management in edge computing systems. Through the implementation of the AdaBoost model on AWS, the customer is able to use machine learning algorithms to dynamically control and manage the use of the resources. The cloud platform offers the required computational resources for prediction and model execution in real time; this is important for computation intensive applications that need quick decisions and minimum response times. This integration makes it possible for edge computing framework to have access to the advanced analyses as well as the effective utilization of resources that enhance the performance and reliability of the framework.

The introduction of the UI interface developed with Flask for the interaction with the API to the cloud-based system enhances its usability. The frontend interface of this created model incorporates the use of Bootstrap to provide users with a platform through which they can input data and get predictions while not having to understand the complex algorithms that are behind the model. This integration becomes useful in the functionality of the system since it makes the system more versatile for use in different contexts and improves the user interaction with the system. All in all, the proposed use of cloud resources, machine learning, and the user-friendly UI create a helpful solution for managing the edge computing resources.

7.3 Applications of this Research in Future

The findings of this project have important implications for the development of edge computing and resource allocation. Thus, this research contributes to the development of new and highly efficient edge computing systems that can be applied to various and time-dependent tasks. This project can be extended in the future as other machine learning methods and fusion techniques are considered in order to improve the resource allocation process. Also, the use of the cloud computing along with edge computing can result in the enhanced and more effective solutions for the new applications like automotive and transportation, smart city, and augmented reality.

The framework proposed in the current project may be generalized and implemented across different sectors given that efficient resource management is paramount. For instance, it can be applied in health care to track and analyze the patients' information in real time or in manufacturing to control and enhance manufacturing strategies. The applications are almost limitless and as edge computing and machine learning technologies develop further, this research serves as a basis for further research and development of better methods of resource allocation and management.

7.4 Practical Challenges

Some real-world issues can be identified when integrating cloud computing with edge computing and machine learning. Another problem is to find and obtain the proper dataset for

training and testing the proposed machine learning model. The dataset must correspond to the edge environment and include a rich set of samples to enhance the model's performance effectiveness. Another problem is the identification of the suitable cloud service provider and the setup of the environment for the particular project. It is important to evaluate each cloud provider because the selection affects the performance, scalability, and cost.

Also, the appropriate rights for different services have to be set up properly, which can prove to be a challenging task especially when it comes to setting up access control and security measures. Some of the concerns include, for instance, how to work with folder locks when accessing EC2 instances locally and challenges with setting permissions through sudo_chmod. At the end, human mistakes that occur during the UI development process can result in functionality problems and may require rather time-consuming debugging and testing. To this end, it is crucial to tackle these challenges when deploying and using cloud-integrated edge computing solutions.

8 Conclusion and Future work

8.1 Conclusion

From this project, it can be concluded that the proposed machine learning algorithm when combined with cloud computing can greatly improve the scheduling of resources in edge computing systems. The present project has described the performance of the AdaBoost fusion model that has been developed and deployed on AWS for the management of resource allocation problems. This success on the AWS EC2 platform relied on the scalability and the computational power of the cloud to boost latency aware applications such as IoT and self-governing cars. The model's compatibility with the cloud infrastructure not only enhanced the computational speed but also assured the required versatility and dependability for real-time analysis.

The experimentation phase revealed that the cloud-based environment was superior at managing the complicated and ever-changing resource management issues compared to the traditional approach. The identified framework used the available AWS services in order to meet the main goals regarding the efficient resource management and low latency. The results obtained in the presented cloud environment suggest that further developments can be built on the successful application of the machine learning models, which means that integrating advanced analytics with the cloud can open new horizons for edge computing.

8.2 Future Work

Further studies should be conducted to investigate the use of other patterns of machine learning and other cloud solutions to improve edge computing resource management. Other cloud services and architectures can be explored, for instance, AWS Lambda for serverless computing which might be more efficient and cheaper for dynamic workloads. Moreover, the use of multi-cloud strategies may provide better reliability and redundancy when resources are spread across the different cloud providers, decreasing the chances of getting a single point of failure and increasing the overall system efficiency.

The future work should be directed towards the improvement of the methods for the integration and interface for the end user to ease the process of use. The use of the latest cloud-based monitoring, and automation tools can assist in efficiently managing the resource and maintaining the models' optimality. Also, the issue of security measures should be considered as well as the real-time data analysis for the improved performance. Thus, the future work can be focused on developing the proposed framework further to create even more effective and versatile approaches for the edge computing applications.

9 References

Aghapour, Z., Sharifian, S., & Taheri, H. (2023). Task offloading and resource allocation algorithm based on deep reinforcement learning for distributed AI execution tasks in IoT edge computing environments. Computer Networks, 223, 109577.

Alfakih, T., Hassan, M. M., Gumaei, A., Savaglio, C., & Fortino, G. (2020). Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA. IEEE Access, 8, 54074-54084.

Deng, X., Li, J., Shi, L., Wei, Z., Zhou, X. and Yuan, J., 2020. Wireless powered mobile edge computing: Dynamic resource allocation and throughput maximization. IEEE Transactions on Mobile Computing, 21(6), pp.2271-2288.

Gao, J., Kuang, Z., Gao, J., & Zhao, L. (2022). Joint offloading scheduling and resource allocation in vehicular edge computing: A two-layer solution. IEEE transactions on vehicular technology, 72(3), 3999- 4009.

https://www.kaggle.com/datasets/omarsobhy14/5g-quality-of-service?select=Quality+of+Service+5G.csv

Hussain, F., Hassan, S.A., Hussain, R. and Hossain, E., 2020. Machine learning for resource management in cellular and IoT networks: Potentials, current solutions, and open challenges. IEEE communications surveys & tutorials, 22(2), pp.1251-1275.

Linder, S., Afzal, S., Bauer, C., Amirpour, H., Prodan, R. and Timmerer, C., 2024, April. VEED: Video Encoding Energy and CO2 Emissions Dataset for AWS EC2 instances. In Proceedings of the 15th ACM Multimedia Systems Conference (pp. 332-338).

Mao, S., He, S. and Wu, J., 2020. Joint UAV position optimization and resource scheduling in space-air-ground integrated networks with mixed cloud-edge computing. IEEE Systems Journal, 15(3), pp.3992-4002.

Qian, L., Wu, Y., Jiang, F., Yu, N., Lu, W. and Lin, B., 2020. NOMA assisted multi-task multi-access mobile edge computing via deep reinforcement learning for industrial Internet of Things. IEEE Transactions on Industrial Informatics, 17(8), pp.5688-5698.

Talaat, F.M., 2022. Effective prediction and resource allocation method (EPRAM) in fog computing environment for smart healthcare system. Multimedia Tools and Applications, 81(6), pp.8235-8258.

Tan, L., Kuang, Z., Zhao, L. and Liu, A., 2021. Energy-efficient joint task offloading and resource allocation in OFDMA-based collaborative edge computing. IEEE Transactions on Wireless Communications, 21(3), pp.1960-1972.

Teoh, Y.K., Gill, S.S. and Parlikad, A.K., 2021. IoT and fog-computing-based predictive maintenance model for effective asset management in Industry 4.0 using machine learning. IEEE Internet of Things Journal, 10(3), pp.2087-2094.

Voron, F., 2023. Building Data Science Applications with RestAPI (Bottle API): Develop, manage, and deploy efficient machine learning applications with Python. Packt Publishing Ltd.

Wang, F., Jiang, D., Qi, S., Qiao, C. and Shi, L., 2021. A dynamic resource scheduling scheme in edge computing satellite networks. Mobile Networks and Applications, 26, pp.597-608.

Wang, T., Lu, B., Wang, W., Wei, W., Yuan, X., & Li, J. (2022). Reinforcement learning-based optimization for mobile edge computing scheduling game. IEEE Transactions on Emerging Topics in Computational Intelligence, 7(1), 55-64.

Xiong, X., Zheng, K., Lei, L. and Hou, L., 2020. Resource allocation based on deep reinforcement learning in IoT edge computing. IEEE Journal on Selected Areas in Communications, 38(6), pp.1133-1146.